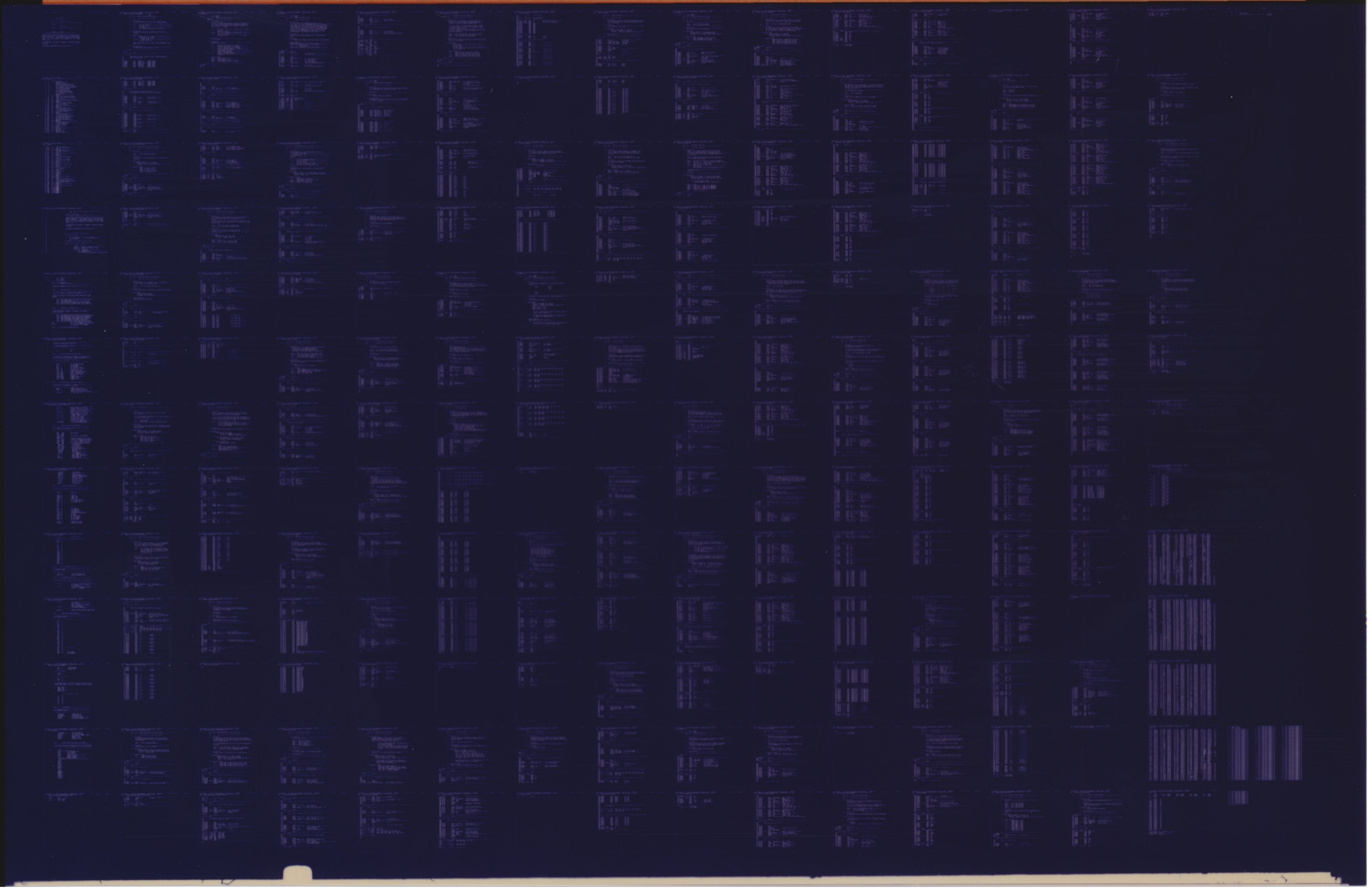
ESKAD·13.2 VAX 11/780 (KA780) EP-ESKAD-DL-13.2 APR 1983
HRDCR TEST STREAM FICHE 1 OF 1 COPYRIGHT@ 77-83
MADE IN USA



IDENTIFICATION

PRODUCT ID: ZZ-ESKAD-13.2

PRODUCT TITLE: ESKAD132 VAX 11/780 HARDCORE TEST STREAM

DECO/DEPO: 13.2

DATE: JANUARY 1983

MAINTAINED BY: VAX DIAGNOSTIC ENGINEERING

COPYRIGHT (C) 1977,1983
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 10754

THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A SINGLE COMPUTER SYSTEM AND MEY BE COPIED ONLY WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

```
     3-    53     "COMMON DEFINITIONS
     3-    53     "       MNEUMONIC DEFINITIONS
     3-    53     "       GLOBAL MACRO CALLS
     3-    53     "       HARDCORE TEST STREAM MACRO CALLS
     3-    53     "       CMPCA AND CMPCAM MODE DEFINITIONS
     3-    53     "       SWITCH (SWR) REGISTER BIT DEFINITIONS
     3-    53     "       SWITCH REGISTER 1 (SWR1) BIT DEFINITIONS
     3-    53     "       CONSOLE ADAPTER REGISTER DEFINITIONS
     3-    53     "       ID BUS REGISTER DEFINITIONS
     3-    53     "       LSI-11 VECTOR DEFINITIONS
     3-    53     "       MISCELLANEOUS DEFINITIONS
     3-    53     "       MODULE AND BUS NAME ASSIGNMENTS
     3-    53     "       LSI-11 REGISTER NAME ASSIGNMENTS
     3-    53     "       FILE NAME CODES
     3-    53     "       CONSOLE ROUTINE ERROR CODES AND DEFINITIONS
     4-    54     SECTION NUMBER 01
     5-    91     T01     CONSOLE ADAPTER REGISTER RESPONSE
     6-   190     T02     CONSOLE "TO ID" REGISTER DATA INTEGRITY
     7-   258     T03     CONSOLE "MCR" REGISTER DATA INTEGRITY
     9-   354     T04     V BUS SELF TEST
    10-   458     T05     CONSOLE "IDCS" REGISTER DATA INTEGRITY
    11-   596     T06     CONSOLE RXDNE AND TXRDY REG DATA INTEGRITY
    12-   679     SECTION NUMBER 02
    12-   679     T07     TXREADY AND RXDONE INTERRUPTS
    13-   805     T08     ID BUS DATA LINES DATA INTEGRITY
    14-   946     T09     CONSOLE CLOCK CONTROL
    15-  1065     T0A     CONSOLE ID CYCLE FUNCTION
    16-  1138     T0B     CONSL FROM ID REG CLK CTRL & DATA INTEG
    17-  1234     SECTION NUMBER 03
    17-  1234     T0C     CONSOLE MAINTENANCE RETURN
    18-  1328     T0D     RXCS REGISTER FROM THE ID BUS SIDE
    19-  1448     T0E     TXCS REGISTER ON THE ID BUS
    20-  1559     T0F     ID BUS REGISTER ADDRESS INTEGRITY
    21-  1701     T10     CIB INITIALIZE FUNCTION
    22-  1840     SECTION NUMBER 04
    22-  1840     T11     CONSOLE REGISTER DUAL ADDRESSING
    23-  1907     T12     WCS DATA REGISTER READ
    24-  1932     T13     INITIALIZE THE CONTROL STORE
    25-  1978     T14     WCS ADDRESS REGISTER DATA INTEGRITY
    26-  2068     T15     WCS ADDRESS REGISTER COUNT LOGIC
    27-  2139     T16     MICRO STACK DATA INTEGRITY
    28-  2229     T17     MICRO STACK DUAL ADDRESSING
    29-  2329     T18     MAINTENANCE RETURN DATA INTEGRITY
    30-  2502     SECTION NUMBER 05
    30-  2502     T19     MAINTENANCE RETURN MICRO STACK INCREMENT
    31-  2547     T1A     MICRO STACK WRITE DISABLE
    32-  2605     T1B     WCS PARITY GENERATORS
    33-  2842     T1C     CS BUS DATA INTEGRITY
    34-  2997     T1D     PCS PARITY CHECKERS
    35-  3129     SECTION NUMBER 06
    35-  3129     T1E     WCS DUAL ADDRESSING 5TH, 6TH, 7TH & 8TH K
    36-  3225     SECTION NUMBER 07
    37-  3264     T1F     WCS DYNAMIC MEMORY TEST
    37-  3354     SECTION NUMBER 08
    38-  3375     T20     NUA BUS DRIVERS
    39-  3435     T21     UBEN FIELD DECODE
    40-  3538     T22     USUB FIELD "CALL FUNCTION"
```

TABLE OF CONTENTS

41- 3650 T23 USUB FIELD ''RETURN''
42- 3713 T24 USUB FIELD ''SELECT SPECIFIER''
43- 3850 T25 UJMP FIELD DATA INTEGRITY
44- 3995 SECTION NUMBER 09
44- 3995 T26 MICRO BREAK REGISTER DATA INTEGRITY
45- 4111 T27 MICRO BREAK COMPARITORS
46- 4272 T28 MICRO ECO
47- 4365 T29 MICRO STACK PUSH WITH CS PARITY ERROR
48- 4501 SECTION NUMBER 0A
48- 4501 T2A INITIALIZE MICRO TRAP
48- 4544 SECTION NUMBER 0B
49- 4578 T2B ID BUS TO DATA PATH INTERFACE
50- 4760 T2C ALU ALEG DATA PATH ''RAMX_Q''
50- 4875 SECTION NUMBER 0C
51- 4907 T2D ALU ALEG DATA PATH ''RAMX_D''
52- 5079 T2E DATA PATH ALU BLEG ''RBMX_Q''
52- 5194 SECTION NUMBER 0D
53- 5226 T2F DATA PATH ALU BLEG ''RBMX_D''
53- 5373 SECTION NUMBER 0E
54- 5408 T30 DATA PATH ALU ARITHMETIC MODE
54- 5713 SECTION NUMBER 0F
55- 5746 T31 DATA PATH ALU ARITHMETIC MODE
55- 5940 SECTION NUMBER 10
56- 5976 T32 ALU CARRY AND MODE CONTROL
57- 6219 T33 DATA PATH KMX
58- 6323 T34 ID BUS WRITE/READ
59- 6433 SECTION NUMBER 11
59- 6433 T35 ''D BYTES'' BRANCH
59- 6666 SECTION NUMBER 12
60- 6700 T36 ALU BRANCH
60- 6994 SECTION NUMBER 13
61- 7035 T37 SCRATCH PAD DATA INTEGRITY
61- 7258 SECTION NUMBER 14
62- 7287 T38 SCRATCH PAD DUAL ADDRESSING
62- 7536 SECTION NUMBER 15
63- 7559 T39 ID MAINT LOCKOUT WHILE CLK RUNNING
64- 7614 T3A RUN BIT IN THE MCS REGISTER
65- 7675 T3B CONSOLE ACK BIT IN THE MCS REGISTER
66- 7732 T3C CONSOLE COMMAND MODE BIT IN THE MCS REG
67- 7809 T3D ACCELERATOR VBUS INTEGRITY
67- 7868 SECTION NUMBER 16
68- 7869 T3E RESERVED FOR FUTURE EXPANSION
68- 7871 SECTION NUMBER 17
69- 7872 T3F RESERVED FOR FUTURE EXPANSION
69- 7874 SECTION NUMBER 18
69- 7876 SECTION NUMBER 19
69- 7878 SECTION NUMBER 1A
69- 7880 SECTION NUMBER 1B
69- 7882 SECTION NUMBER 1C
69- 7884 SECTION NUMBER 1D
69- 7886 SECTION NUMBER 1E
69- 7888 SECTION NUMBER 1F

```
                        .TITLE  VAX 11/780 MICRO DIAGNOSTIC HARDCORE TEST STREAM
                        .IDENT  /V13.0/
                ;
                ; COPYRIGHT (C) 1977,1983
                ; DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
                ;
                ; THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE  ONLY ON A SINGLE
                ; COMPUTER  SYSTEM AND  MAY BE  COPIED ONLY WITH  THE INCLUSION OF THE
                ; ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,
                ; MAY NOT BE PROVIDED OR  OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
                ; EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE
                ; TERMS.  TITLE TO AND  OWNERSHIP OF THE  SOFTWARE  SHALL AT ALL TIMES
                ; REMAIN IN DEC.
                ;
                ; THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
                ; AND SHOULD  NOT BE CONSTRUED  AS A COMMITMENT  BY DIGITAL  EQUIPMENT
                ; CORPORATION.
                ;
                ; DEC ASSUMES  NO  RESPONSIBILITY  FOR  THE USE OR  RELIABILITY OF ITS
                ; SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.
                ;


                ;++
                ; FACILITY: MICRO DIAGNOSTIC HARDCORE MONITOR
                ;
                ; FUNCTIONAL DESCRIPTION:
                ;
                ;       THIS FILE CONTAINS THE TESTS FOR THE HARDCORE LOGIC OF THE
                ;       PDP 11/780 PROCESSOR. IT IS LOADED AND EXECUTED BY THE
                ;       HARDCORE MONITOR.
                ;
                ; ENVIRONMENT:  MICRO DIAGNOSTIC HARDCORE MONITOR
                ;
                ; AUTHOR:  DONALD W. MONROE,  CREATION DATE:  11-OCT-1977
                ;
                ; MODIFIED BY:  DON MONROE : MAY 1979
                ;
                ;               VERSION 11 - DELETED THE TEST OF THE SLGW CONSTANT ROM.
                ;               VERSION 11.2 - MODIFIED TO SUPPORT 4K OF WCS
                ;               VERSION 11.3 - ADDED A TEST FOR THE FPA VBUS.
                ;               VERSION 13.0 - APRIL 1981
                ;                       ADDED FILE NAME AND VERSION TO DIRECTORY.
                ;               VERSION 13.1 - DECEMBER 1982
                ;                       FIXED PROBLEM WHEN ONE M8238 (2K OF WCS) WAS INSTALLED
                ;                       ONLY THE FIRST 1K OF THE 2K MODULE WAS GETTING TESTED.
                ;--
```

```
                        .NLIST  MD,CND,BIN
                        .LIST   MC,ME
                        .MCALL  EQUATE
000000                  EQUATE  HARDCORE

                .SBTTL  ''COMMON DEFINITIONS
                .SBTTL  ''      MNEUMONIC DEFINITIONS
                ;*****************************************************************************
                ;+
                ; FOLLOWING ARE COMMON DEFINITIONS OF MNEUMONICS USED BY ALL OF THE
                ; PROGRAMS THAT EXECUTE OUT OF THE LSI-11.
                ;-
                ;
                .SBTTL  ''      GLOBAL MACRO CALLS
                ;*****************************************************************************
                ;+
                ; THE FOLLOWING ".MCALL'S" ARE GLOBAL MACRO ASSIGNMENTS. THESE MACRO'S ARE
                ; USED BY ALL 4 MONITORS, THE PARSER, AND THE DIRECTORY FILE.
                ;
                ; SOME OF THESE MACRO'S ARE DEFINED IN THE CONSOLE MACRO PACGAGE "STRMAC"
                ; THEREFORE, THAT MACRO FILE MUST BE MERGED WITH THIS MACRO FILE BEFORE
                ; ASSEMBLY.
                ;-
                ;*****************************************************************************
                        .MCALL  T$INIT,T$WRIT,T$READ,F$OPEN,F$READ,LOADCO,CONVERT,CONABORT
                        .MCALL  LDCNSL,GETMDM,ENCTRLC
                        .MCALL  STARS,COMTAGS,OPENFILE,READOVR,RETURN,RESET$,ASSEMBLE
                        .MCALL  DONE,RDIDREG,SBCCLOCK,DONEM,FILL,CALLFAILCHAIN,$CODDF
                        .MCALL  MES,TYPES,TYPED,TYPE,TYPEMOD,RINGBELL,GETUPC,TYPESECTNO
                        .MCALL  TYPEERR,CALLMICMON,LOADWCS,STSCLOCK,LOADID,MTPS,MFPS,TYPEB
                ;
                ;
                ;
                .SBTTL  ''      HARDCORE TEST STREAM MACRO CALLS
                ;*****************************************************************************
                ;+
                ; THE FOLLOWING MACRO CALLS ARE USED EXCLUSIVELY BY THE HARDCORE TEST
                ; STREAM. THEY ARE ONLY ASSEMBLED IF THE ARGUMENT TO THE "EQUATE"
                ; MACRO IS NON BLANK
                ;-
                ;*****************************************************************************
                        .MCALL  NEWTST,$$NEWTST,NEWOVR,FORCEOVR,INITIALIZE,MASK,HEXTST,DUMMY
                        .MCALL  LOOP,$$LOOP,ENDLOOP,ERLOOP,IFERROR,CMPCA,CMPCAM,$$ERRLOOP
                        .MCALL  NOOP,READVB,BLKMIC,CLOCK,TSTVB,LDIDREG,SKIP,SPAGEN,$$SKIP
                        .MCALL  $$$SK,TESTDAT,ENDDAT,CMPPCSV,$ENDDAT,HEX3,HEX33,MOVE
                        .MCALL  CMPCAD,CMPCMD,READID,CHKPNT,REPORT,FLTONE,FLTZRO,KMXGEN
                        .MCALL  LOADCA,ENDHC,$$ENDHC,FETCH,$$REPORT,VBUSG,$$FETCH
                        .MCALL  HEX1,HEX11,HEX2,HEX22,HEX4,HEX44,HEX5,HEX55,HEX6,HEX66
                        .MCALL  SETVEC,SUBTEST,RESETC,SETPSW,FILL,$DIR,TYPSIZE,SKIPERROR
$TN=1                                   ; INIT THE TEST NUMBER FOR THE TEST STREAM
$SN=1                                   ; INIT THE SECTION NUMBER FOR THE TEST STREAM
$SECTOR=0                               ; INIT THE RELATIVE SECTOR NUMBER FOR
                                        ; THE DIRECTORY OF THE TEST STREAM
                ;
                .SBTTL  ''      CMPCA AND CMPCAM MODE DEFINITIONS
                ;*****************************************************************************
```

'' CMPCA AND CMPCAM MODE DEFINITIONS

```
;+
; FOLLOWING ARE THE TWO MODE DEFINITIONS FOR THE "CMPCA", "CMPCAD",
; "CMPCAM", AND "CMPCMD" PSEUDO INSTRUCTIONS.
;
; IF MORE MODES ARE REQUIRED, THE BRANCH TABLE (IN THE HARDCORE MONITOR)
; WILL HAVE TO HAVE MORE ENTRYS PUT IN IT.
;-
;*****************************************************************************
;
        EQ.=    0               ; BEQ
        NE.=    4               ; BNE
;
;
.SBTTL  ''      SWITCH (SWR) REGISTER BIT DEFINITIONS
;*****************************************************************************
;+
; FOLLOWING ARE THE DEFINITIONS OF THE 16 BITS OF THE SOFTWARE SWITCH
; REGISTER. BITS<5:0> ARE READ WRITE BY COMMAND FROM THE MICRO MONITOR.
; BITS 7 AND 6 ARE READ AND CLEAR ONLY FROM THE MICRO MONITOR.
;
; ALL OTHER BITS ARE TRANSPARENT TO THE OPERATOR.
;-
;*****************************************************************************
;
        HALTD=  1               ; HALT ON ERROR DETECTION
        HALTI=  2               ; HALT ON ERROR ISOLATION
        LOOP=   4               ; LOOP ON ERROR
        NER=    10              ; NO ERROR REPORT
        BELL=   20              ; BELL ON EVERY 6 ERRORS
        ERABT=  40              ; GO TO NEXT TEST AFTER ERROR
        LOSS=   100             ; LOOP ON SPECIAL SECTION
        LOST=   200             ; LOOP ON SPECIAL TEST
        SINST=  400             ; HARDCORE SINGLE INSTRUCTION FLAG
        FLPY2=  1000            ; MA780 FLOPPY (MOUNTED) FLAG
        FLPY3=  2000            ; FLOPPY 2 (MOUNTED) FLAG
        FLPY4=  3000            ; MS780-E FLOPPY (MOUNTED) FLAG
        FLPYMSK=3000            ; MASK FOR FLOPPY FIELD
        CONT=   4000            ; CONTINUE FLAG
        KEYQUE= 10000           ; KEYBOARD ILLEGAL CHARACTER
        KEYERR= 20000           ; KEYBOARD ERROR FLAG
        CTRLC=  40000           ; CONTROL C FLAG
        COM=    100000          ; COMMAND MODE FLAG
;
;
.SBTTL  ''      SWITCH REGISTER 1 (SWR1) BIT DEFINITIONS
;*****************************************************************************
;+
; FOLLOWING ARE THE BIT DEFINITIONS OF SOFTWARE SWITCH REGISTER 1 (SWR1).
; THESE BITS ARE ALL TRANSPARENT TO THE OPERATOR.
;-
;*****************************************************************************
        HARDC=  1               ; HARDCORE (EXECUTING) FLAG
        RUNFLG= 2               ; DIAGNOSE COMMAND HAS BEEN USED
        B1FULL= 4               ; BUFFER 1 FULL FLAG. USED IN THE
                                ; DOUBLE BUFFERED ROUTINE IN THE GO CHAIN
        CLKFST= 10              ; SET CLOCK FAST FLAG. SET OR CLEARED BY THE OPERATOR
        CLKSLO= 20              ; SET CLOCK SLOW FLAG.
```

```
''       SWITCH REGISTER 1 (SWR1) BIT DEFINITIONS

                B2INUSE=40              ; BUFFER 2 IN USE FLAG. USED BY THE DOUBLE
                                        ; BUFFER ROUTINE IN THE GO CHAIN
                DIRERR= 100             ; DIRECTORY ERROR FLAG. SET BY THE "DIRECTORY"
                                        ; PROGRAM IF AN ERROR WAS DETECTED
                B2FULL= 200             ; BUFFER 2 FULL FLAG. USED BY THE DOUBLE
                                        ; BUFFER ROUTINE IN THE GO CHAIN
                B1INUSE=400             ; BUFFER 1 IN USE FLAG. USED BY THE DOUBLE
                                        ; BUFFER ROUTINE IN THE GO CHAIN
                DICMD=  1000            ; DIAGNOSE COMMAND FLAG. SET WHEN A
                                        ; "DIAGNOSE" COMMAND IS USED
                MIC1FL= 2000            ; GO CHAIN FILE # 1 FLAG. USED BY THE
                                        ; DIRECTORY SEARCH PROGRAM
                MIC2FL= 4000            ; GO CHAIN FILE # 2 FLAG. USED BY THE
                                        ; DIRECTORY SEARCH PROGRAM.
                FPA=    10000           ; FPA PRESENT FLAG. SET BY THE GO CHAIN
                                        ; MONITOR OR THE COMMAND PARSER.
                TSTSPAN=20000           ; A TEST SPAN HAS BEEN SPECIFIED
                SCTSPAN=40000           ; A SECTION SPAN HAS BEEN SPECIFIED

        ;
        ;
        .SBTTL  ''      CONSOLE ADAPTER REGISTER DEFINITIONS
        ;*****************************************************************8
        ;+
        ; THE FOLLOWING ARE THE ADDRESS ASSIGNMENTS AND THE BIT DEFINITIONS
        ; OF THE CONSOLE ADAPTER REGISTERS.
        ;-
        ;*********************************************************************
        ;
                ROM0=   173000          ; ROM LOCATION 0
                ROM1=   173002          ; ROM LOCATION 2
                SPARE1= 173004
                IDDATLO=173006          ; LOW 16 BITS OF ID DATA REGISTER
                IDDATHI=173010          ; HIGH 16 BITS OF ID DATA REGISTER
                SPARE2= 173012
                RXDNE=173014            ; RECEIVER CONTROL AND STATUS REGISTER
                TXRDY=  173016          ; TRANSMITTER CONTROL AND STATUS REGISTER
                TOIDLO= 173020          ; LO 16 BITS OF TRANSMITTER DATA BUFFER
                TOIDHI= 173022          ; HIGH 16 BITS OF TRANSMITTER DATA BUFFER
                FMIDLO= 173024          ; LO 16 BITS OF RECEIVER DATA BUFFER
                FMIDHI= 173026          ; HIGH 16 BITS OF RECEIVER DATA BUFFER
                IDCS=   173030          ; ID BUS CONTROL AND STATUS REGISTER
                  IDMAINT=200           ;   ID MAINTENANCE BIT
                  IDWRITE=100           ;   ID BUS WRITE BIT
                  IDCYCLE=100000        ;   ID BUS CYCLE BIT
                CONMCR= 173032          ; MACHINE CONTROL REGISTER
                  INIT= 10000           ;   CPU INITIALIZE BIT
                  MNTRTN=2000           ;   MAINTENANCE RETURN ENABLE BIT
                  UPC12=1000            ;   FORCE UPC 12 BIT
                  CLRUWRD=200           ;   ROM NOP BIT
                  SOMM=100              ;   STOP ON MICRO MATCH ENABLE BIT
                  CLKSTPD=40            ;   CLOCK STOPPED BIT
                  FR1=  20              ;   CLOCK FREQUENCY SELECT BIT 1
                  FR0=  10              ;   CLOCK FREQUENCY SELECT BIT 0
                  STS=  4               ;   ENABLE SINGLE TIME STATE BIT
                  SBC=  2               ;   ENABLE SINGLE BUS CYCLE BIT
                  PROCEED=1             ;   CLOCK PROCEED BIT
                CONMCS= 173034          ; MACHINE CONTROL AND STATUS REGISTER
```

''   CONSOLE ADAPTER REGISTER DEFINITIONS

```
                FLPYON=10000            ;   FLOPPY ON BIT
                CONCM=1000              ;   CONSOLE COMMAND MODE BIT
                CPURUN=400              ;   CPU RUN BIT
                CONACK=200              ;   CONSOLE ACKNOWLEDGE BIT
                RDYIE=100               ;   RECEIVER INTERRUPT ENABLE BIT
                DNEIE=40                ;   TRANSMITTER INTERRUPT ENABLE
              VBCTRL= 173036            ; V BUS CONTROL REGISTER
                CCPT0=200               ;   TIME STATE CPT0 BIT
                CCPT1=100               ;   TIME STATE CPT1 BIT
                CCPT2=40                ;   TIME STATE CPT2 BIT
                CCPT3=20                ;   TIME STATE CPT2 BIT
                SLFTST=4                ;   V BUS SELF TEST BIT
                VBLOAD=2                ;   V BUS LOAD BIT
                VBCLK=1                 ;   V BUS CLOCK BIT
;
.SBTTL  ''      ID BUS REGISTER DEFINITIONS
;*****************************************************************************
;+
; FOLLOWING ARE THE MNEUMONICS ASSIGNED TO THE ID BUS REGISTERS.
;-
;*****************************************************************************
;
        IBDAT=  00              ; IBUF DATA
        IBTOD=  01              ; TIME OF DAY CLOCK
;
        CONID=  03              ; SYSTEM ID
        CONRXS= 04              ; CONSOLE RXCS
        CONRXD= 05              ; CONSOLE RXDB
        CONTXS= 06              ; CONSOLE TXCS
        CONTXD= 07              ; CONSOLE TXDB
        RWDQ=   10              ; WRITE Q REG, READ D REG
        IBNIN=  11              ; NEXT INTERVAL REGISTER
        IBCLKS= 12              ; CLOCK CONTROL AND STATUS
        IBICT=  13              ; IBUF INTERVAL COUNT
        CES=    14
        VECT=   15
        SIR=    16
        PSL=    17
        TBDAT=  20              ; TBUF DATA
;
        TBER0=  22              ; TBUF ERROR REG 0
        TBER1=  23              ; TBUF ERROR REG 1
        ACC0=   24              ; ACCELERATOR REG 0
        ACC1=   25              ; ACCELERATOR REG 1
        ACCMNT= 26              ; ACCELERATOR MAINTENANCE REG
        ACCST=  27              ; ACCELERATOR STATUS REGISTER
        SBISILO=30              ; SBI SILO
        SBIERR= 31              ; SBI ERROR REG
        SBITO=  32              ; SBI TIMEOUT ADDRESS
        SBIFLT= 33              ; SBI FAULT/STATUS
        SBISCM= 34              ; SBI SILO COMAPRATOR
        SBIMAT= 35              ; SBI MAINTENANCE
        SBICP=  36              ; SBI CACHE PARITY
;
        USCSTK=40               ; SEQUENCER MICRO STACK
        USCBRK=41               ; SEQUENCER MICRO BREAK
        USCADR=42               ; SEQUENCER WCS ADDRESS
```

```
''       ID BUS REGISTER DEFINITIONS

                USCDAT=43               ; SEQUENCER WCS DATA

;
; THE FOLLOWING REGISTERS ARE THE TEMPA AND TEMPB REGISTERS
;
                P0BR=   44              ;
                P1BR=   45
                SBR=    46
;
                KSP=    50
                ESP=    51
                SSP=    52
                USP=    53
                ISP=    54
                FPDA=   55
                D.SV=   56
                Q.SV=   57
                TEMP0=  60
                TEMP1=  61
                TEMP2=  62
                TEMP3=  63
                TEMP4=  64
                TEMP5=  65
                TEMP6=  66
                TEMP7=  67
                TEMP8=  70
                TEMP9=  71
                PCBB=   72
                SCBB=   73
                P0LR=   74
                P1LR=   75
                SLR=    76
;
.SBTTL  ''      LSI-11 VECTOR DEFINITIONS
;*****************************************************************************
;+
; THE FOLLOWING MNEUMONICS ARE THE DEFINITIONS FOR THE LSI-11 TRAP
; AND INTERRUPT VECTORS.
;-
;*****************************************************************************
;
        TRAPVEC=34              ; ''TRAP'' INSTRUCTION VECTOR
        TXVEC=  300             ; TRANSMITTER INTERRUPT VECTOR
        RXVEC=  304             ; RECEIVER INTERRUPT VECTOR
;
.SBTTL  ''      MISCELLANEOUS DEFINITIONS
;*****************************************************************************
;+
; FOLLOWING ARE SOME MISCELLANEOUS DEFINITIONS USED IN THE HARDCORE TESTS.
;-
;*****************************************************************************
;
        IDREGLO=-1              ; USED AFTER A ''READID'' PSEUDO INSTRUCTION TO SPECIFY
                                ; THE CONTENTS OF LOCATION ''IDDATLO''
                                ; AS THE ARGUMENT
        IDREGHI=1               ; USED AFTER A ''READID'' PSEUDO INSTRUCTION
                                ; TO SPECIFY THE CONTENTS OF LOCATION
                                ; ''IDDATHI'' AS THE ARGUMENT
```

```
''       MISCELLANEOUS DEFINITIONS

                TPCINIT=34              ; FIRST ADDRESS (RELATIVE) OF EACH TEST
                                        ; STREAM OVERLAY.
                                        ; NOTE: IF THE LENGTH OF THE DISPATCH
                                        ; TABLE IS CHANGED, THIS DEFINITION
                                        ; MUST ALSO BE CHANGED.
                ITSTPTR=4               ; FIRST ADDRESS (RELATIVE) OF THE TEST TABLE
                                        ; IN EACH TEST STREAM OVERLAY.
;
                RADOCT= 10              ; RADIX OCTAL CODE FOR CONSOLE CONVERT ROUTINE
                RADHEX= 20              ; RADIX HEX CODE FOR CONSOLE CONVERT ROUTINE
;
;
.SBTTL  ''      MODULE AND BUS NAME ASSIGNMENTS
;*****************************************************************************
;+
; THE FOLLOWING DEFINITIONS ARE USED BY THE ''TYPMOD'' ROUTINE IN THE
; MICRO DIAGNOSTIC MONITOR.
;-
;*****************************************************************************
;
                CIB=    0
                USC=    1
                WCS=    2
                PCS=    3
                DAP=    4
                DBP=    10
                DCP=    5
                DDP=    6
                DEP=    7
                CEH=    11
                ICL=    12
                CAM=    13
                CDM=    14
                TBM=    15
                SBL=    16
                SBH=    17
                IRC=    20
                IDP=    21
                MSB=    22
                MCN=    23
                MDT=    24
                MAY=    25
                CLK=    26
                TRS=    27
                FNM=    30
                FMH=    31
                FML=    32
                FAD=    33
                FCT=    34
                MAY6=   35      ; MAY WITH 16K CHIP
                MPI=    36
                MPC=    37
                MPS=    40
                MAT=    41
                WCS2K=  42      ; 2K WCS MODULE
                MSBE=   43      ; MSB FOR MA780-E
                BYL=    44      ; LOWER CONTROLLER
```

```
''        MODULE AND BUS NAME ASSIGNMENTS

                BYU=    45      ; UPPER CONTROLLER
                MAY4=   46      ; 1 MEGABYTE ARRAY
                MAY8=   47      ; 4 MEGABYTE ARRAY
        ;
        ; START OF BUS NAMES
        ;
                CSBUS=  50
                IDBUS=  51
                VBUS=   52
        ;
        ; START OF ADAPTER NAMES
        ;
                UBA=    53
                MBA=    54
                DRA=    55
                CIA=    56
;
; THE FOLLOWING OFFSET DEFINITIONS ARE OFFSETS INTO THE RAD50 LIST FOR
; THE ABOVE MODULE NAMES. IF THE LIST IS CHANGED, THESE OFFSETS MUST BE
; CHANGED. THESE OFFSETS ARE USED BY THE TYPE MODULE ROUTINE IN ESKAB.
;
                BUSOFF= CSBUS * 2
                M4KOFF= MAY * 2
                M6KOFF= MAY6 * 2
                M64KOF= MAY4 * 2
                M256KO= MAY8 * 2
                ADAOFF= UBA * 2
;
;
.SBTTL  ''      LSI-11 REGISTER NAME ASSIGNMENTS
;
                R0=     %0
                R1=     %1
                R2=     %2
                R3=     %3
                R4=     %4
                R5=     %5
                R6=     %6
                R7=     %7
                SP=     %6
                PC=     %7
;
;
.SBTTL  ''      FILE NAME CODES
;*****************************************************************************
;+
; THE FOLLOWING CODES ARE USED BY THE "OPEN FILE" ROUTINE IN THE
; MICRO DIAGNOSTIC MONITOR.
;-
;*****************************************************************************
;
                HCMONITOR=0             ; HARDCORE MONITOR
                TESTSTREAM=2            ; HARDCORE TEST STREAM
                GOCHAINMONITOR=4        ; GO CHAIN MONITOR
                GOCHA1=6                ; GO CHAIN FILE NUMBER 1 (FLOPPY 1)
                PARSER=10               ; MICRO DIAGNOSTIC PARSER
                GOCHA2=12               ; GO CHAIN FILE NUMBER 2 (FLOPPY 2)
```

```
''       FILE NAME CODES

                        DIRECTORY=14            ; DIRECTORY SEARCH FILE
                        FAILCHAINMONITOR=16     ; FAIL CHAIN MONITOR
                        FCHAI1=20               ; FAIL CHAIN FILE NUMBER 1 (FLOPPY 1)
                        FCHAI2=22               ; FAIL CHAIN FILE NUMBER 2 (FLOPPY 2)
                        MPGOCH=24               ; MA780 GO CHAIN
                        MPFC=26                 ; MA780 FAIL CHAIN
                        MSGOCH=30               ; MS780-E GO CHAIN
                        MSFC=32                 ; MS780-E FAIL CHAIN
                ;
                ;
                ;
                .SBTTL  ''      CONSOLE ROUTINE ERROR CODES AND DEFINITIONS
                ;*****************************************************************************
                ;+
                ; THE FOLLOWING ARE ERROR CODE DEFINITIONS AND EMT DEFINITIONS DEFINED
                ; BY MIKE HARE THAT ARE USED TO COMMUNICATE WITH THE CONSOLE ROUTINES.
                ;-
                ;*****************************************************************************
                ;
        000000          $CODDF
                        ;FLOPPY AND TERMINAL ERROR CODES
                        $FER=1          ;FLOPPY HARDWARE ERROR
                        $FNF=2          ;FILE NOT FOUND
                        $FNR=3          ;FLOPPY QUEUE FULL
                        $FOR=4          ;FLOPPY SECTOR # OUT OF LEGAL RANGE
                        $TBSY=5         ;NO NODE FOR REQUEST
                        $TCTC=6         ;CONTROL-C INPUTTED
                        $TER=7          ;TERMINAL HARDWARE DETECTED ERROR
                        ;USER SERVICE EMT CODE DEFINITIONS
                        ;THESE CODES MUST BE IN SYNC WITH THE EMT SERVICE MODULE
                        TINIT=0
                        TWRITE=1
                        TREAD=2
                        OPENFL=3
                        READSC=4
                        WRITSC=5
                        LOADCN=6
                        CNVERT=7
                        RADGET=10
                        OPNFL1=11
                        TYP1=12
                        TYP2=13
                        LCANWC=14
                        RMWRON=15
                        LCWRON=16
                        TMERTR=17
                        R$SET=20
                        LDCONS=21
                        MDMTYP=22
                        CHKSWI=23
                        TSTMFG=24
```

;;   CONSOLE ROUTINE ERROR CODES AND DEFINITIONS

```
     54 000000            NEWOVR  1,,,ESKAD,13,1
        000024            HEX6    \1
                  .SBTTL  SECTION NUMBER 01
        000000            .PSECT  OVR01
     55                   .LIST   ME
```

```
     91          .SBTTL  T01     CONSOLE ADAPTER REGISTER RESPONSE
                 ;;*****************************************************************
                 ;++
                 ;TEST   01      CONSOLE ADAPTER REGISTER RESPONSE
                 ;
                 ; TEST DESCRIPTION
                 ;       THIS TEST READS AND WRITES ALL THE CONSOLE ADAPTER REGISTERS
                 ;       TO ENSURE THE ADDRESS DECODE ON THE CIB IS OPERATING CORRECTILY.
                 ;
                 ;       SUBTST 1 - READ ALL THE REGISTERS THAT ARE IMPLEMENTED AND INSURE THAT
                 ;                  THEY DON'T TRAP TO 4.
                 ;       SUBTST 2 - READ THOSE REGISTERS THAT ARE NOT IMPLEMENTED AND ENSURE THAT
                 ;                  THEY TRAP TO 4.
                 ;       SUBTST 3 - WRITE THOSE REGISTERS THAT ARE READ ONLY AND THOSE THAT ARE
                 ;                  NOT IMPLEMENTED AND ENSURE THAT THEY TRAP TO 4.
                 ;
                 ; LOGIC DESCRIPTION
                 ;       THIS TEST CHECKS THE LOGIC THAT DECODES THE CONSOLE ADAPTER
                 ;       REGISTER SPACE ADDRESSES AND RETURNS THE SYNC TO THE LSI-11, ON THE
                 ;       CIB MODULE.
                 ;
                 ; ERROR DESCRIPTION
                 ;       SUBTST 1 - UNEXPECTED TRAP TO 4 MEANS THAT A CIB REGISTER DID NOT
                 ;                  RESPOND WHEN IT SHOULD HAVE.
                 ;       SUBTST 2 - IF AN ERROR OCCURS, IT MEANS THAT A REGISTER DID NOT TIME
                 ;                  OUT WHEN IT SHOULD HAVE.
                 ;       SUBTST 3 - IF AN ERROR OCCURS, IT MEANS THAT A REGISTER DID NOT TIME
                 ;                  OUT WHEN IT SHOULD HAVE.
                 ;
                 ;       FOR ALL THREE SUBTESTS, THE DATA THAT IS TYPED IS MEANINGLESS. THE
                 ;       ERROR NUMBER WILL INDICATE WHICH CONSOLE ADAPTER REGISTER WAS UNDER
                 ;       TEST.
                 ;
                 ; SYNC POINT DESCRIPTION
                 ;       THE EASIEST WAY TO DEBUG THIS TEST IS TO TOGGLE IN A MOV INSTRUCTION
                 ;       THAT READS OR WRITES THE REGISTER THAT FAILED.
                 ;--
                 ;;*****************************************************************
     000034      T01:
     95
     96 000040           SUBTEST
                 ;////////////////////////////////////////////////////////////////
     000040      T01S1:
     97          ;+
     98          ; FIRST CHECK THAT A TRAP TO 4 DOES NOT OCCUR ON THOSE REGISTERS THAT
     99          ; CAN BE READ. THIS SUBTEST REPORTS A FAILURE BY TYPING AN UNEXPECTED
    100          ; TRAP TO 4 ERROR MESSAGE.
    101          ;-
    102
    103 000042           CMPCA   ,ROM0,T00L1     ; ADDRESS 163000
    104 000054           CMPCA   ,ROM1,T00L1     ; ADDRESS 163002
    105 000066           CMPCA   ,IDDATLO,T00L1  ; ADDRESS 163006
    106 000100           CMPCA   ,IDDATHI,T00L1  ; ADDRESS 163010
    107 000112           CMPCA   ,RXDNE,T00L1    ; ADDRESS 163014
    108 000124           CMPCA   ,TXRDY,T00L1    ; ADDRESS 163016
    109 000136           CMPCA   ,TOIDLO,T00L1   ; ADDRESS 163020
    110 000150           CMPCA   ,TOIDHI,T00L1   ; ADDRESS 163022
```

```
    111 000162          CMPCA   ,FMIDLO,TOOL1   ; ADDRESS 163024
    112 000174          CMPCA   ,FMIDHI,TOOL1   ; ADDRESS 163026
    113 000206          CMPCA   ,IDCS,TOOL1     ; ADDRESS 163030
    114 000220          CMPCA   ,CONMCR,TOOL1   ; ADDRESS 163032
    115 000232          CMPCA   ,CONMCS,TOOL1   ; ADDRESS 163034
    116 000244          CMPCA   ,VBCTRL,TOOL1   ; ADDRESS 163036
    117
    118 000256          SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
        000256  T01S2:
    119         ;+
    120         ; NOW CHECK READS TO NON-RESPONDING REGISTERS
    121         ; THE GOOD AND BAD DATA THAT GETS TYPED OUT IS MEANINGLESS.
    122         ; IF AN ERROR OCCURS, IT MEANS A REGISTER DID NOT TIME OUT.
    123         ;-
    124
    125 000260          ERLOOP
    126 000262          SETVEC  4               ; SET THE CPU TRAP VECTOR
    127 000266          CMPCA   ,SPARE1,TOOL1   ; ADDRESS 163004
    128 000300          IFERROR ,CIBT00         ; REGISTER DID NOT TIMEOUT
    129 000306          ERLOOP
    130 000310          SETVEC  4
    131 000314          CMPCA   ,SPARE2,TOOL1   ; ADDRESS 163012
    132 000326          IFERROR ,CIBT00         ; REGISTER DID NOT TIMEOUT
    133 000334          SKIP    SUBTST3         ; GO TO NEXT SUBTEST
    134
    135 000340  CIBT00: REPORT  <CIB>
    136
    137 000350          SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
        000350  T01S3:
    138         ;+
    139         ; NOW CHECK WRITES TO NON RESPONDING REGISTERS
    140         ;-
    141
    142 000352          ERLOOP
    143 000354          SETVEC  4
    144 000360          LOADCA  ROM0,TOOL1      ; ADDRESS 163000
    145 000370          IFERROR ,CIBT00         ; REGISTER DID NOT TIMEOUT
    146 000376          ERLOOP
    147 000400          SETVEC  4
    148 000404          LOADCA  ROM1,TOOL1      ; ADDRESS 163002
    149 000414          IFERROR ,CIBT00         ; REGISTER DID NOT TIMEOUT
    150 000422          ERLOOP
    151 000424          SETVEC  4
    152 000430          LOADCA  SPARE1,TOOL1    ; ADDRESS 163004
    153 000440          IFERROR ,CIBT00
    154 000446          ERLOOP
    155 000450          SETVEC  4
    156 000454          LOADCA  SPARE2,TOOL1    ; ADDRESS 163012
    157 000464          IFERROR ,CIBT00
    158
    159 000472          TESTDAT
                ;-------------------------------------------------------------------------------
    160 000476  TOOL1:  .WORD
    161 000500          ENDDAT
                ;-------------------------------------------------------------------------------
```

```
                .SBTTL  T02     CONSOLE "TO ID" REGISTER DATA INTEGRITY
                ;*****************************************************************
                ;++
                ;TEST   02      CONSOLE "TO ID" REGISTER DATA INTEGRITY
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST FLOATS A ONE AND A ZERO THRU THE "TO ID " REGISTER.
                ;
                ;       SUBTST 1 - FLOAT A ZERO THRU THE REGISTER
                ;       SUBTST 2 - FLOAT A ONE THRU THE REGISTER
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE LOAD FUNCTION OF THE TO ID REGISTER, THE
                ;       "TO ID" INPUT TO THE Q DATA MULTIPLEXOR, AND THE Q DATA BUS ON THE
                ;       CIB MODULE.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED CONTENTS OF "TO ID" REGISTER
                ;             RECEIVED CONTENTS OF "TO ID" REGISTER
                ;             LOOP COUNT -- INDICATES WHICH BIT POSITION THE ONE OR ZERO IS
                ;               CURRENTLY IN, I.E. 1=BIT 0, 2=BIT 1, ETC.
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC00--"TO ID LOW" IS LOADED
                ;                  SYNC01--"TO ID HIGH" IS LOADED
                ;                  SYNC02--"TO ID REG IS READ
                ;       SUBTST 2 - SYNC03--TO ID REG LOW IS LOADED
                ;                  SYNC04--TO ID REG HIGH IS LOADED
                ;                  SYNC05--TI ID REG IS READ
                ;--
                ;*****************************************************************
    000500      T02:
194 000504              INITIALIZE
195
196 000506              SUBTEST
                ;/////////////////////////////////////////////////////////////////
    000506      T02S1:
197             ;+
198             ; FLOAT A ZERO THRU THE REGISTER
199             ;-
200
201 000510              LOOP    I,1,32
202 000522              FLTZRO  CTMP1,I         ; GENERATE THE TEST PATTERN
203 000530              ERLOOP
204 000532      SYNC00: LOADCA  TOIDLO,CTMP1    ; LOAD THE LOW 16 BITS
205 000542      SYNC01: LOADCA  TOIDHI,CTMP1+2  ; AND THE UPPER 16 BITS
206 000552      SYNC02: CMPCAD  EQ,TOIDLO,CTMP1 ; READ REG BACK AND CHECK IT
207 000576              IFERROR 51,CIB1         ; BIT STUCK IN "TO ID" REGISTER
208 000604              ENDLOOP I               ; CONTINUE
209
210 000610              SUBTEST
                ;/////////////////////////////////////////////////////////////////
    000610      T02S2:
211             ;+
212             ; FLOAT A ONE THRU THE REGISTER
213             ;-
214
```

```
    215 000612            LOOP    I,1,32
    216 000624            FLTONE  CTMP1,I          ; GENERATE THE TEST PATTERN
    217 000632            ERLOOP
    218 000634  SYNC03:   LOADCA  TOIDLO,CTMP1     ; LOAD THE DATA PATTERN (LO 16 BITS)
    219 000644  SYNC04:   LOADCA  TOIDHI,CTMP1+2   ; AND THE HIGH 16 BITS
    220 000654  SYNC05:   CMPCAD  EQ,TOIDLO,CTMP1  ; READ THE DATA BACK AND CHECK IT
    221 000700            IFERROR 50,CIB1          ; BIT STUCK IN "TO ID" REG
    222 000706            ENDLOOP I                ; CONTINUE
    223
    224 000712            SKIP
    225 000716  CIB1:     REPORT  <CIB>
    226
    227 000726            TESTDAT
                ;-------------------------------------------------------------------------------
    228 000732  CTMP1:    .WORD   0,0              ; WORKING LOCATION FOR DATA PATTERNS
    229 000736            ENDDAT
                ;-------------------------------------------------------------------------------
    230
    231
    254
```

```
    258          .SBTTL  T03     CONSOLE "MCR" REGISTER DATA INTEGRITY
                 ;;*****************************************************************************
                 ;++
                 ;TEST   03      CONSOLE "MCR" REGISTER DATA INTEGRITY
                 ;
                 ; TEST DESCRIPTION
                 ;       THIS TEST CHECKS THAT THE "MACHINE CONTROL" (MCR) REGISTER
                 ;       RESOPNDS PROPERLY. ONLY CERTAIN BITS CAN BE SET OR CLEARED FROM
                 ;       THE LSI-11, SO ONLY TWO PATTERNS ARE TESTED.
                 ;
                 ;       SUBTEST 1 - CHECK THAT "CLOCK STOPPED" BIT IS SET.
                 ;       SUBTEST 2 - CHECK THE OTHER BITS IN THE REGISTER.
                 ;
                 ; LOGIC DESCRIPTION
                 ;       THIS TEST CHECKS THE Q BUS INTERFACE TO THE MACHINE CONTROL REGISTER
                 ;       ON THE CIB MODULE AND THE "CLOCK STOPPED" SIGNAL FROM THE CLK MODULE.
                 ;
                 ; ERROR DESCRIPTION
                 ;       DATA: EXPECTED CONTENTS OF MCR REG
                 ;             RECEIVED CONTENTS OF MCR REG
                 ;             LOOP COUNT -- INDICATES WHICH PATTERN IS BEING USED, I.E.
                 ;               1=PATTERN 9554, 2=PATTERN 028A
                 ;
                 ; SYNC POINT DESCRIPTION
                 ;       ADDRESS SYNC06--MCR REG IS LOADED
                 ;       ADDRESS SYNC07--MCR REGISTER IS READ
                 ;--
                 ;;*****************************************************************************
    000736       T03:
    262 000742           INITIALIZE
    263
    264 000744           SUBTEST
                 ;//////////////////////////////////////////////////////////////////////////////
    000744       T03S1:
    265          ;+
    266          ; CHECK THAT THE "CLOCK STOPPED" BIT IS SET.
    267          ;-
    268
    269 000746           CMPCAM  ,CONMCR,T03L1,,T03L1    ; CHECK "CLOCK STOPPED" BIT SET
    270 000762           IFERROR ,CIB3A                  ; IF NOT, REPORT CLK AND CIB
    271 000770           SKIP    SUBTEST                 ; GO TO NEXT SUBTEST
    272
    273 000774   CIB3A:  REPORT  <CLK,CIB>
    274
    275 001004           SUBTEST
                 ;//////////////////////////////////////////////////////////////////////////////
    001004       T03S2:
    276          ;+
    277          ; NOW CHECK THE OTHER BITS
    278          ;-
    279
    280 001006           LOOP    I,1,2           ; GOING TO CHECK TWO PATTERNS
    281 001020           ERLOOP
    282 001022   SYNC06: LOADCA  CONMCR,CTMP2,I  ; LOAD THE TEST PATTERN
    283 001032   SYNC07: CMPCA   EQ,CONMCR,CTMP3,I ; READ IT BACK AND CHECK IT
    284 001044           IFERROR 54,CIB3         ; BIT(S) STUCK IN MCR REGISTER
    285 001052           ENDLOOP I               ; CONTINUE
```

T03 CONSOLE "MCR" REGISTER DATA INTEGRITY

```
001056          LOADCA  CONMCR,CTMP98   ; CLEAR ALL BITS BUT SBC
001066          SKIP
001072  CIB3:   REPORT  <CIB>

001102          TESTDAT
        ;-------------------------------------------------------------------------------

        ;+
        ; THE FOLLOWING IS THE TEST DATA USED FOR THIS TEST:
        ;-

001106  CTMP2:  .WORD   112524          ; HLTREQ, INIT, MNTRTN, INTR DSABL, SOMM
        ;   HEXDATA     9554
                                        ; FR1, AND STS,
001110          .WORD   1212            ; UPC12,CLRUWRD,FR0, AND SBC, MNTRTN
        ;   HEXDATA     28A

        ;+
        ; FOLLOWING IS THE EXPECTED DATA FOR THIS TEST:
        ;-

001112  CTMP3:  .WORD   112564          ; HLTREQ,INIT,MNTRTN,INTR DSABL,SOMM
        ;   HEXDATA     9574
                                        ; CLKSTPD,FR1, AND STS
001114          .WORD   1252            ; UPC12,CLRUWRD,CLKSTPD,FR0, AND SBC
        ;   HEXDATA     2AA
001116  CTMP98: .WORD   SBC
001120  T03L1:  .WORD   CLKSTPD
001122          ENDDAT
        ;-------------------------------------------------------------------------------

```

```
    354          .SBTTL  T04     V BUS SELF TEST
                 ;;*****************************************************************************
                 ;++
                 ;TEST   04      V BUS SELF TEST
                 ;
                 ; TEST DESCRIPTION
                 ;       THIS TEST LOADS THE VBUS CHANNELS WITH ALL ZERO'S AND THEN ALL
                 ;       ONE'S, WITH THE SELF TEST BIT, AND CHECKS THAT THE CORRECT
                 ;       DATA IS READ BACK.
                 ;
                 ;       FIRST, THE LOAD SIGNAL IS ISSUED TO CHECK THAT IT GETS FROM THE CONSOLE
                 ;       TO THE FIRST VBUS CHIP ON ANY CHANNEL. THIS IS DONE BY TESTING FOR
                 ;       A NON ZERO BIT ON ANY CHANNEL.
                 ;
                 ;       SUBTST 1 - LOAD ALL ZERO'S, THEN ISSUE A LOAD PULSE, AND CHECK FOR
                 ;                  A NON ZERO BIT.
                 ;       SUBTST 2 - LOAD ALL ZERO'S, THEN READ THE BUS BACK AND CHECK.
                 ;       SUBTST 3 - LOAD ALL ONE'S, THEN READ THE BUS BACK AND CHECK.
                 ;
                 ; LOGIC DESCRIPTION
                 ;       THIS TEST CHECKS THAT THE V BUS CLOCK, LOAD, AND SELF TEST SIGNALS
                 ;       ARE TRANSMITTED FROM THE CIB MODULE TO ALL THE APPROPRIATE MODULES,
                 ;       AND THAT THE SERIAL INPUT/OUTPUT LINES OF ALL THE V BUS CHIPS ARE
                 ;       WORKING CORRECTLY.
                 ;
                 ; ERROR DESCRIPTION
                 ;       DATA: EXPECTED CONTENTS OF V BUS CONTROL REGISTER
                 ;             RECEIVED CONTENTS OF V BUS CONTROL REGISTER
                 ;             LOOP COUNT -- INDICATES WHICH BIT OF THE V BUS FAILED,
                 ;               I.E. 1=BIT 0, 2=BIT 1, 3=BIT 2, ETC.
                 ;
                 ; SYNC POINT DESCRIPTION
                 ;       SUBTST 1 - SYNC98--ISSUE THE LOAD PULSE
                 ;       SUBTST 2 - SYNC12--LOAD THE V BUS WITH A ZERO
                 ;                  SYNC13--CHECK THE CURRENT BIT OF THE V BUS
                 ;                  SYNC14--SHIFT THE V BUS
                 ;       SUBTST 3 - SYNC15--LOAD THE V BUS WITH A ONE
                 ;                  SYNC16--CHECK THE CURRENT BIT OF THE V BUS
                 ;                  SYNC17--CLOCK THE V BUS
                 ;--
                 ;;*****************************************************************************
    001122       T04:
    358 001126           INITIALIZE
    359
    360 001130           SUBTEST
                 ;//////////////////////////////////////////////////////////////////////////////
    001130       T04S1:
    361              ;+
    362              ; LOAD THE V BUS WITH ALL ZEROS AND CHECK THAT THE LOAD PULSE LOADS IN ATLEAST
    363              ; ONE DATA BIT.
    364              ;-
    365
    366 001132           LOOP    I,1,112         ; LOAD THE VBUS WITH ALL ZERO'S
    367 001144           LOADCA  VBCTRL,TMP5B    ; CLOCK THE BUS
    368 001154           ENDLOOP I               ; CONTINUE
    369
    370 001160           LOADCA  VBCTRL,TMP5C    ; SET THE LOAD BIT
```

```
    371 001170  SYNC98: LOADCA  VBCTRL,TMP5A     ; CLEAR THE LOAD BIT
    372 001200           CMPCAM  NE,VBCTRL,MSK5A,,TMP5A ; CHECK FOR A NON ZERO BIT
    373 001214           IFERROR ,CIB5A           ; V BUS DID NOT LOAD
    374
    375 001222           SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
        001222  T04S2:
    376                  ;+
    377                  ; LOAD THE V BUS WITH ALL ZERO'S AND CHECK THAT IT GOT LOADED
    378                  ;-
    379
    380 001224           ERLOOP
    381 001226           LOOP    I,1,112          ; LOAD THE VBUS WITH ALL ZERO'S
    382 001240  SYNC12:  LOADCA  VBCTRL,TMP5B     ; CLOCK THE BUS
    383 001250           ENDLOOP I                ; CONTINUE
    384
    385                  ;+
    386                  ; NOW CHECK THAT ALL ZERO'S GOT LOADED
    387                  ;-
    388
    389 001254           LOOP    I,1,112          ; 112 BITS IN THE LONGEST CHANNEL
    390 001266  SYNC13:  CMPCAM  EQ,VBCTRL,MSK5A,,TMP5A ; MAKE THE CHECK
    391 001302           IFERROR ,CIB5A           ; VBUS CHANNEL FAILED
    392 001310  SYNC14:  LOADCA  VBCTRL,TMP5B     ; GET THE NEXT BIT
    393 001320           ENDLOOP I                ; CONTINUE
    394
    395 001324           SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
        001324  T04S3:
    396                  ;+
    397                  ; NOW CHECK THAT ALL ONES CAN BE LOADED
    398                  ;-
    399
    400 001326           ERLOOP
    401 001330           LOOP    I,1,112          ; LOAD THE ONE'S
    402 001342  SYNC15:  LOADCA  VBCTRL,TMP5A+2   ; ...
    403 001352           ENDLOOP I                ; CONTINUE
    404
    405 001356           LOOP    I,1,112          ; GOING TO CHECK THE ONE'S
    406 001370  SYNC16:  CMPCAM  EQ,VBCTRL,MSK5A,,MSK5A ; ...
    407 001404           IFERROR ,CIB5A           ; VBUS BIT FAILED
    408 001412  SYNC17:  LOADCA  VBCTRL,TMP5A+2   ; CLOCK THE BUS
    409 001422           ENDLOOP I                ; CONTINUE
    410 001426           SKIP
    411
    412 001432  CIB5A:   REPORT  <CIB,VBUS>
    413
    414 001442           TESTDAT
                ;-------------------------------------------------------------------------------
    415 001446  TMP5A:   .WORD   0,SLFTST+VBCLK
    416 001452  TMP5B:   .WORD   VBCLK
    417 001454  TMP5C:   .WORD   VBLOAD
    418 001456  MSK5A:   .WORD   37400
    419                ;   HEXDATA     3F00
    420 001460           ENDDAT
                ;-------------------------------------------------------------------------------
    421
```

```
    458          .SBTTL  T05     CONSOLE "IDCS" REGISTER DATA INTEGRITY
                 ;;*****************************************************************************
                 ;++
                 ;TEST   05      CONSOLE "IDCS" REGISTER DATA INTEGRITY
                 ;
                 ; TEST DESCRIPTION
                 ;       THIS TEST CHECKS THE DATA INTEGRITY OF THE ID CONTROL REGISTER
                 ;       (IDCS). THIS IS DONE BY FIRST CHECKING THAT THE ID MAINTENANCE
                 ;       AND CYCLE BITS ARE NOT STUCK, AND THEN FLOATING A ONE AND A
                 ;       ZERO THRU THE LOWER BYTE (WITH ID MAINTENANCE ON) AND
                 ;       CHECKING THAT THE UPPER BYTE REFLECTS THE LOWER BYTE.
                 ;
                 ;       SUBTST 1 - CHECK THAT BITS 7 AND 15 CAN BE SET AND CLEARED.
                 ;       SUBTST 2 - FLOAT A ZERO AND A ONE THRU THE LOWER BYTE AND CHECK
                 ;                  THAT THE UPPER BYTE IS THE COMPLIMENT OF THE LOWER.
                 ;                  ALSO CHECK ADDRESS BITS 0,1, AND 2 ON THE V BUS TO
                 ;                  ENSURE THAT THE "RIGHT" SIDE OF THE ID BUS IS
                 ;                  FUNCTIONING CORRECTLY.
                 ;
                 ;       NOTE: THE ADDRESS BITS <13:8> AND <5:0> ARE OPPOSITE POLARITY.
                 ;
                 ; LOGIC DESCRIPTION
                 ;       THIS TEST CHECKS THE Q BUS INTERFACE TO THE ID CONTROL AND STATUS
                 ;       REGISTER ON THE CIB MODULE, THE ID BUS ADDRESS LINES, AND THE
                 ;       ID BUS ADDRESS MULTIPLEXOR ON THE ICL MODULE.
                 ;
                 ; ERROR DESCRIPTION
                 ;       DATA: EXPECTED CONTENTS OF THE IDCS REGISTER
                 ;             RECEIVED CONTENTS OF THE IDCS REGISTER
                 ;             LOOP COUNT -- INDICATES WHICH PATTERN IS CURRENTLY UNDER TEST,
                 ;               I.E. 1=3E81, 2=3D82, 3=3B84, ETC.
                 ;
                 ; SYNC POINT DESCRIPTION
                 ;       SUBTST 1 - SYNC08--BITS 7 AND 15 ARE WRITTEN TO THE IDCS
                 ;                  SYNC09--BITS 7 AND 15 ARE READ FROM THE IDCS
                 ;       SUBTST 2 - SYNC0A--LOW BYTE OF IDCS IS WRITTEN
                 ;                  SYNC0B--IDCS IS READ AND CHECKED.
                 ;--
                 ;;*****************************************************************************
    001460       T05:
    462 001464           INITIALIZE
    463
    464 001466           SUBTEST
                 ;//////////////////////////////////////////////////////////////////////////////
    001466       T05S1:
    465          ;+
    466          ; FIRST CHECK THAT BITS 7 AND 15 SET AND CLEAR PROPERLY.
    467          ;-
    468
    469 001470           LOOP    I,1,2
    470 001502           ERLOOP
    471 001504   SYNC08: LOADCA  IDCS,CTMP4,I     ; LOAD TEST PATTERN
    472 001514   SYNC09: CMPCAM  EQ,IDCS,CMSK1,,CTMP4,I ; READ AND CHECK THE DATA
    473 001530           IFERROR 55,CIBX4         ; BIT STUCK IN IDCS REGISTER
    474 001536           ENDLOOP I                ; CONTINUE
    475 001542           SKIP    SUBTEST
    476
```

```
    477 001546  CIBX4:  REPORT  <CIB>
    478
    479 001556          SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
        001556  T05S2:
    480         ;+
    481         ; NOW FLOAT A ONE AND A ZERO THRU THE LOW BYTE AND CHECK THAT THE
    482         ; HIGH BYTE ALSO CHANGES TO THE OPPOSITE OF THE LOW BYTE
    483         ;-
    484
    485 001560          LOOP    I,1,14
    486 001572          ERLOOP
    487 001574  SYNCOA: LOADCA  IDCS,CTMP5,I    ; LOAD THE TEST PATTERN
    488 001604  SYNCOB: CMPCA   EQ,IDCS,CTMP5,I ; READ AND CHECK THE REGISTER
    489 001616          IFERROR 56,CIB4         ; BIT STUCK IN IDCS OR ID BUS ADDRESS LINES
    490 001624          TSTVB   T04L1,I         ; CHECK BITS<2:0> ON THE "RIGHT" SIDE
    491 001632          IFERROR ,CIB4A          ; REPORT ICL OR SBH
    492 001640          ENDLOOP I               ; CONTINUE
    493 001644          SKIP
    494 001650  CIB4:   TSTVB   T04L1,I         ; CHECK BITS<2:0> ON THE "RIGHT" SIDE
    495 001656          CHKPNT  ,CIB4A          ; CALL OUT ICL AND SBH IF THEY ARE BAD
    496 001664          REPORT  <CIB,ICL,IDBUS> ; OTHERWISE CALL OUT THESE THREE
    497 001674  CIB4A:  REPORT  <ICL,SBH>
    498
    499 001704          TESTDAT
                ;-------------------------------------------------------------------------------
    500 001710  CTMP4:  .WORD   0,100200        ; MAINTENANCE AND CYCLE BITS
    501         ;   HEXDATA    0,8080
    502 001714  CTMP5:  .WORD   37201,36602,35604,33610,27620,17640,77700 ; FLT ONE PATTERNS
    503         ;   HEXDATA     3E81, 3D82, 3B84, 3788, 2F90, 1FA0, 7FC0
    504 001732          .WORD   40776,41375,42373,44367,50357,60337,00277 ; FLT ZERO PATTERNS
    505         ;   HEXDATA    41FE,  42FD, 44FB, 48F7, 50EF, 60DF, 00BF
    506 001750  CMSK1:  .WORD   100377
    507         ;   HEXDATA    80FF
    508 001752  T04L1:  .WORD   3
    509 001754          VBUSG   5,1,1           ; ID ADDR 2 L
    510 001756          VBUSG   5,2,1           ; ID ADDR 1 L
    511 001760          VBUSG   5,7,0           ; ID ADDR 0 L
    512 001762          .WORD   3
    513 001764          VBUSG   5,1,1           ; ID ADDR 2 L
    514 001766          VBUSG   5,2,0           ; ID ADDR 1 L
    515 001770          VBUSG   5,7,1           ; ID ADDR 0 L
    516 001772          .WORD   3
    517 001774          VBUSG   5,1,0           ; ID ADDR 2 L
    518 001776          VBUSG   5,2,1           ; ID ADDR 1 L
    519 002000          VBUSG   5,7,1           ; ID ADDR 0 L
    520 002002          .WORD   3
    521 002004          VBUSG   5,1,1           ; ID ADDR 2 L
    522 002006          VBUSG   5,2,1           ; ID ADDR 1 L
    523 002010          VBUSG   5,7,1           ; ID ADDR 0 L
    524 002012          .WORD   3
    525 002014          VBUSG   5,1,1           ; ID ADDR 2 L
    526 002016          VBUSG   5,2,1           ; ID ADDR 1 L
    527 002020          VBUSG   5,7,1           ; ID ADDR 0 L
    528 002022          .WORD   3
    529 002024          VBUSG   5,1,1           ; ID ADDR 2 L
    530 002026          VBUSG   5,2,1           ; ID ADDR 1 L
```

!05 CONSOLE "IDCS" REGISTER DATA INTEGRITY

```
531 002030          VBUSG   5,7,1           ; ID ADDR 0 L
532 002032          .WORD   3
533 002034          VBUSG   5,1,1           ; ID ADDR 2 L
534 002036          VBUSG   5,2,1           ; ID ADDR 1 L
535 002040          VBUSG   5,7,1           ; ID ADDR 0 L
536
537
538 002042          .WORD   3
539 002044          VBUSG   5,1,0           ; ID ADDR 2 L
540 002046          VBUSG   5,2,0           ; ID ADDR 1 L
541 002050          VBUSG   5,7,1           ; ID ADDR 0 L
542 002052          .WORD   3
543 002054          VBUSG   5,1,0           ; ID ADDR 2 L
544 002056          VBUSG   5,2,1           ; ID ADDR 1 L
545 002060          VBUSG   5,7,0           ; ID ADDR 0 L
546 002062          .WORD   3
547 002064          VBUSG   5,1,1           ; ID ADDR 2 L
548 002066          VBUSG   5,2,0           ; ID ADDR 1 L
549 002070          VBUSG   5,7,0           ; ID ADDR 0 L
550 002072          .WORD   3
551 002074          VBUSG   5,1,0           ; ID ADDR 2 L
552 002076          VBUSG   5,2,0           ; ID ADDR 1 L
553 002100          VBUSG   5,7,0           ; ID ADDR 0 L
554 002102          .WORD   3
555 002104          VBUSG   5,1,0           ; ID ADDR 2 L
556 002106          VBUSG   5,2,0           ; ID ADDR 1 L
557 002110          VBUSG   5,7,0           ; ID ADDR 0 L
558 002112          .WORD   3
559 002114          VBUSG   5,1,0           ; ID ADDR 2 L
560 002116          VBUSG   5,2,0           ; ID ADDR 1 L
561 002120          VBUSG   5,7,0           ; ID ADDR 0 L
562 002122          .WORD   3
563 002124          VBUSG   5,1,0           ; ID ADDR 2 L
564 002126          VBUSG   5,2,0           ; ID ADDR 1 L
565 002130          VBUSG   5,7,0           ; ID ADDR 0 L
566 002132          ENDDAT
                ;-------------------------------------------------------------------------------
567
```

```
    596          .SBTTL  T06     CONSOLE RXDNE AND TXRDY REG DATA INTEGRITY
                 ;;*****************************************************************
                 ;++
                 ;TEST   06      CONSOLE RXDNE AND TXRDY REG DATA INTEGRITY
                 ;
                 ; TEST DESCRIPTION
                 ;       THIS TEST CHECKS THE DONE AND READY FLAGS IN THE RECEIVER
                 ;       AND TRANSMITTER CONTROL REGISTERS. THIS IS DONE BY WRITING
                 ;       A ONE AND A ZERO TO BIT 7 OF THE REGISTERS AND CHECKING THAT THE
                 ;       BIT SET CORRECTLY.
                 ;
                 ;       SUBTST 1 - CHECK THE RECEIVER DONE BIT
                 ;       SUBTST 2 - CHECK THE TRANSMITTER READY BIT
                 ;
                 ; LOGIC DESCRIPTION
                 ;       THIS TEST CHECKS THE Q BUS INTERFACE TO THE RECEIVER DONE AND
                 ;       TRANSMITTER READY BITS ON THE CIB MODULE.
                 ;
                 ; ERROR DESCRIPTION
                 ;       DATA: EXPECTED CONTENTS OF RECEIVER OR TRANSMITTER STATUS REG
                 ;             RECEIVED CONTENTS OF RECEIVER OR TRANSMITTER STATUS REG
                 ;             LOOP COUNT -- INDICATES WHICH PATTERN IS CURRENTLY UNDER
                 ;               TEST, I.E. 1=0080, 2=0000
                 ;
                 ; SYNC POINT DESCRIPTION
                 ;       SUBTST 1 - SYNC0C--RXCS REG IS LOADED
                 ;                  SYNC0D--RXCS REGISTER IS READ
                 ;       SUBTST 2 - SYNC0E--TXCS REG IS LOADED
                 ;                  SYNC0F--TXCS REGISTER IS READ
                 ;--
                 ;;*****************************************************************
    002132       T06:
    600 002136           INITIALIZE
    601
    602 002140           SUBTEST
                 ;/////////////////////////////////////////////////////////////////
    002140       T06S1:
    603          ;+
    604          ; FIRST CHECK THE RECEIVER DONE BIT
    605          ;-
    606
    607 002142           LOOP    I,1,2
    608 002154           ERLOOP
    609 002156   SYNC0C: LOADCA  RXDNE,CTMP7,I   ; LOAD TEST PATTERN INTO RECEIVER REG
    610 002166   SYNC0D: CMPCA   EQ,RXDNE,CTMP7,I ; READ AND CHECK THE DATA
    611 002200           IFERROR 57,CIB5         ; BIT STUCK IN RECEIVER REGISTER
    612
    613 002206           SUBTEST
                 ;/////////////////////////////////////////////////////////////////
    002206       T06S2:
    614          ;+
    615          ; NOW CHECK THE TRANSMITTER READY BIT
    616          ;-
    617
    618 002210           ERLOOP
    619 002212   SYNC0E: LOADCA  TXRDY,CTMP7,I   ; LOAD DATA PATTERN INTO TRANSMITTER REG
    620 002222   SYNC0F: CMPCA   EQ,TXRDY,CTMP7,I ; READ AND CHECK THE REGISTER
```

```
    621 002234          IFERROR 60,CIB5         ; BIT STUCK IN TRANSMITTER REG
    622 002242          ENDLOOP I               ; CONTINUE
    623 002246          SKIP
    624 002252  CIB5:   REPORT  <CIB>
    625
    626 002262          TESTDAT
                ;-------------------------------------------------------------------------------
    627 002266  CTMP7:  .WORD   200,0,0
    628         ;   HEXDATA    0080,0000,0000
    629 002274          ENDDAT
                ;-------------------------------------------------------------------------------
    630
```

```
                .SBTTL  SECTION NUMBER 02
000000                  .PSECT  OVR02
                .SBTTL  T07     TXREADY AND RXDONE INTERRUPTS
                ;;*****************************************************************
                ;++
                ;TEST   07      TXREADY AND RXDONE INTERRUPTS
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THAT THE RX DONE AND TX READY BITS ALONG WITH THE
                ;       ASSOCIATED INTERRUPT ENABLES IN THE MCS REGISTER WILL INTERRUPT THE
                ;       LSI-11. THIS IS DONE BY EITHER SETTING THE UNEXPECTED OR EXPECTED
                ;       TRAP CATCHER TO THE INTERRUPT VECTOR, CREATING THE INTERRUPT, AND
                ;       CHECKING TO SEE IF IT OCCURRED.
                ;
                ;       SUBTST 1 - CHECK THAT TX READY DOES NOT INTERRUPT WHEN THE TX READY
                ;                  BIT IS SET.
                ;       SUBTST 2 - CHECK THAT TX READY CAUSES AN INTERRUPT WHEN IT TRANSITIONS
                ;                  FROM A ONE TO A ZERO.
                ;       SUBTST 3 - CHECK THAT RX DONE DOES NOT INTERRUPT WHEN IT IS SET.
                ;       SUBTST 4 - CHECK THAT RX DONE CAUSES AN INTERRUPT WHEN IT TRANSITIONS
                ;                  FROM A ONE TO A ZERO.
                ;       SUBTST 5 - CHECK THAT THE TX READY INTERRUPT HAS PRIORITY OVER THE
                ;                  RX DONE INTERRUPT.
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE INTERRUPT FUNCTIONS OF THE DC003 CHIP
                ;       ON THE CIB MODULE.
                ;
                ; ERROR DESCRIPTION
                ;       THE ERROR DATA IS MEANINGLESS. FOLLOWING IS A DESCRIPTION OF THE POSSIBLE
                ;       ERRORS FOR EACH SUB TEST:
                ;
                ;       SUBTST 1 - UNEXPECTED INTERRUPT MEANS THAT THE TXREADY BIT
                ;                  BEING SET DID NOT INHIBIT THE INTERRUPT.
                ;       SUBTST 2 - AN ERROR REPORT MEANS THAT THE TXREADY BIT GOING FROM
                ;                  A ONE TO A ZERO DID NOT CAUSE AN INTERRUPT.
                ;       SUBTST 3 - UNEXPECTED INTERRUPT MEANS THAT THE RXDONE BIT BEING SET
                ;                  DID NOT INHIBIT THE INTERRUPT.
                ;       SUBTST 4 - AN ERROR REPORT MEANS THAT THE TRANSITION OF THE RXDONE BIT
                ;                  DID NOT CAUSE AN INTERRUPT.
                ;       SUBTST 5 - UNEXPECTED INTERRUPT MEANS THAT THE RXDONE INTERRUPT OCCURRED
                ;                  INSTEAD OF THE TXREADY INTERRUPT.
                ;                  AN ERROR MESSAGE MEANS THAT THE TXREADY INTERRUPT DID
                ;                  NOT OCCUR.
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNCA2--TXIE BIT IS SET
                ;       SUBTST 2 - SYNCA3--TXREADY GOES FROM ONE TO ZERO
                ;       SUBTST 3 - SYNCA4--RXIE BIT IS SET
                ;       SUBTST 4 - SYNCA5--RXDONE GOES FROM ONE TO ZERO
                ;       SUBTST 5 - SYNCA6--PROCESSOR PRIORITY IS LOWERED TO ZERO
                ;--
                ;;*****************************************************************
000034          T07:
683 000040              INITIALIZE
684
685 000042              SUBTEST
```

```
                ;////////////////////////////////////////////////////////////////////////////////
        000042  T07S1:
    686         ;+
    687         ; CHECK THAT TXREADY ON A ONE DOES NOT INTERRUPT
    688         ;-
    689
    690 000044          ERLOOP
    691 000046          SETPSW  0                   ; LOWER THE PSW PRIORITY
    692 000052          LOADCA  TXRDY,T36L1         ; SET THE TX READY BIT
    693 000062  SYNCA2: LOADCA  CONMCS,T36L2        ; SET THE INTERRUPT ENABLE BIT
    694         ;+
    695         ; UNEXPECTED TRAP WILL OCCUR HERE IF THIS TEST FAILS
    696         ;-
    697 000072          LOADCA  CONMCS,T36L3        ; CLEAR THE INTERRUPT ENABLE
    698
    699 000102          SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
        000102  T07S2:
    700         ;+
    701         ; CHECK THAT THE TRANSITION OF THE TXREADY BIT CAUSES AN INTERRUPT
    702         ;-
    703
    704 000104          ERLOOP
    705 000106          SETPSW  0                   ; LOWER THE PSW PRIORITY LEVEL
    706 000112          LOADCA  TXRDY,T36L1         ; SET THE TX READY BIT
    707 000122          LOADCA  CONMCS,T36L2        ; SET THE INTERRUPT ENABLE
    708 000132          SETVEC  TXVEC               ; SET THE INTERRUPT VECTOR
    709 000136  SYNCA3: LOADCA  TXRDY,T36L1+2       ; CLEAR THE READY BIT, SHOULD CAUSE INTERRUPT
    710 000146          IFERROR ,CIB36              ; INTERRUPT DID NOT OCCUR
    711 000154          LOADCA  CONMCS,T36L3        ; CLEAR THE INTERRUPT ENABLE
    712
    713 000164          SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
        000164  T07S3:
    714         ;+
    715         ; CHECK THAT THE RXDONE BEING SET DOES NOT INTERRUPT
    716         ;-
    717
    718 000166          ERLOOP
    719 000170          SETPSW  0                   ; SET PROCESSOR PRIORITY TO ZERO
    720 000174          LOADCA  RXDNE,T36L1         ; SET THE DONE BIT
    721 000204  SYNCA4: LOADCA  CONMCS,T36L4        ; SET THE INTERRUPT ENABLE
    722         ;+
    723         ; UNEXPECTED INTERRUPT WILL OCCUR HERE IF THIS SUBTEST FAILS
    724         ;-
    725 000214          LOADCA  CONMCS,T36L3        ; CLEAR THE INTERRUPT ENABLE
    726
    727 000224          SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
        000224  T07S4:
    728         ;+
    729         ; CHECK THAT THE TRANSITION OF THE RXDONE BIT CAUSES AN INTERRUPT
    730         ;-
    731
    732 000226          ERLOOP
    733 000230          SETPSW  0                   ; LOWER THE PROCESSOR PRIORITY
    734 000234          LOADCA  RXDNE,T36L1         ; SET THE DONE BIT
```

```
000244          LOADCA  CONMCS,T36L4     ; SET THE INTERRUPT ENABLE
000254          SETVEC  RXVEC             ; SET THE INTERRUPT VECTOR
000260  SYNCA5: LOADCA  RXDNE,T36L1+2     ; CLEAR THE DONE BIT, CAUSES INTERRUPT
000270          IFERROR ,CIB36            ; DONE BIT DID NOT CAUSE INTERRUPT
000276          LOADCA  CONMCS,T36L3      ; CLEAR THE INTERRUPT ENABLE

000306          SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
000306  T07S5:
                ;+
                ; CHECK THAT THE READY INTERRUPT HAS PRIORITY OVER THE DONE INTERRUPT
                ;-

000310          SETPSW  340               ; ENSURE INTERRUPTS ARE LOCKED OUT
000314          ERLOOP
000316          LOADCA  TXRDY,T36L1+2     ; CLEAR THE TXREADY BIT
000326          LOADCA  RXDNE,T36L1+2     ; AND THE RXDONE BIT
000336          SETVEC  TXVEC             ; SET THE TRANSMITTER VECTOR
000342          SETPSW  0                 ; SET PRIORITY AT ZERO
000346  SYNCA6: LOADCA  CONMCS,T36L5      ; SET BOTH INTERRUPT ENABLE BITS, TX SHOULD INTRPT
                ;+
                ; AN UNEXPECTED INTERRUPT MEANS THAT THE RECEIVER INTERRUPTED
                ;-
000356          IFERROR ,CIB36            ; CHECK THAT THE INTERRUPT OCCURRED
000364          LOADCA  CONMCS,T36L3      ; CLEAR THE INTERRUPT ENABLES
000374          SKIP

000400  CIB36:  REPORT  <CIB>

000410          TESTDAT
                ;-------------------------------------------------------------------------------
000414  T36L1:  .WORD   200,0
000420  T36L2:  .WORD   FLPYON+RDYIE
000422  T36L3:  .WORD   FLPYON
000424  T36L4:  .WORD   FLPYON+DNEIE
000426  T36L5:  .WORD   FLPYON+RDYIE+DNEIE
000430          ENDDAT
                ;-------------------------------------------------------------------------------
```

```
    805          .SBTTL  T08     ID BUS DATA LINES DATA INTEGRITY
                 ;;******************************************************************************
                 ;++
                 ;TEST   08      ID BUS DATA LINES DATA INTEGRITY
                 ;
                 ; TEST DESCRIPTION
                 ;       THIS TEST CHECKS THE DATA INTEGRITY OF THE ID BUS DATA LINES.
                 ;       THIS IS DONE BY FLOATING A ZERO AND A ONE THRU THE TO ID REGISTER
                 ;       WITH THE MAINTENANCE MODE BIT SET, AND READING THE ID DATA
                 ;       REGISTER.
                 ;       FIRST, AN ALTERNATING ONE'S AND ZERO'S PATTERN IS PLACED IN THE "TO ID"
                 ;       REGISTER, THE IDCS IS SET TO DO A WRITE WITH THE ID MAINT BIT CLEAR,
                 ;       AND THE ID BUS IS CHECKED TO ENSURE THE "TO ID" REGISTER IS NOT ENABLED
                 ;       ONTO THE ID BUS.
                 ;
                 ;       SUBTST 1 - CHECK THE ID BUS WITH ID MAINT CLEAR
                 ;       SUBTST 2 - FLOAT A ZERO THRU THE ID BUS DATA LINES
                 ;       SUBTST 3 - FLOAT A ONE THRU THE ID BUS DATA LINES
                 ;
                 ; LOGIC DESCRIPTION
                 ;       THIS TEST CHECKS THAT THE "TO ID" REGISTER IS GATED ONTO THE ID
                 ;       BUS AND THE ID DATA INPUTS TO THE QDATA MUX ON THE
                 ;       CIB MODULE. IT ALSO CHECKS THAT NO OTHER MODULES ARE PUTTING RANDOM
                 ;       DATA ON THE ID BUS.
                 ;
                 ; ERROR DESCRIPTION
                 ;       DATA: EXPECTED VALUE OF THE ID DATA LINES
                 ;             RECEIVED VALUE OF THE ID DATA LINES
                 ;             LOOP COUNT -- INDICATES THE BIT POSITION OF THE FLOATING PATTERN,
                 ;               I.E. 1=BIT 0, 2=BIT 1, 3=BIT 2, ETC.
                 ;
                 ; SYNC POINT DESCRIPTION
                 ;       SUBTST 1 - SYNC97--ID BUS IS READ AND CHECKED
                 ;       SUBTST 2 - SYNC10--ID DATA IS READ AND CHECKED
                 ;       SUBTST 3 - SYNC11--ID DATA IS READ AND CHECKED
                 ;--
                 ;;******************************************************************************
    000430       T08:
809 000434           SUBTEST
                 ;////////////////////////////////////////////////////////////////////////////////
    000434       T08S1:
810              ;+
811              ; FIRST CHECK THE ID BUS WITH ID MAINT CLEAR
812              ; THE EXPECTED AND RECEIVED DATA SHOULD NOT BE THE SAME.
813              ;-
814
815 000436           INITIALIZE
816 000440           LOADCA  TOIDLO,T05L2    ; PUT DATA PATTERN IN TO ID REG
817 000450           LOADCA  TOIDHI,T05L2    ; ...
818 000460           LOADCA  IDCS,T05L3      ; PUT ADDRESS OF TO ID REG IN IDCS
819 000470           CLOCK   3               ; GO TO CPT150
820 000474   SYNC97: CMPCAD  NE,IDDATLO,T05L2 ; CHECK ID BUS OTHER THAN DATA PATTERN
821 000520           IFERROR ,XCIB6A         ; ID MAINT CLEAR DOES NOT DISABLE MAINT FCT
822 000526           SKIP    SUBTST
823
824 000532   XCIB6A: REPORT  <ICL,CIB>
825
```

```
826
827 000542                SUBTEST
                    ;////////////////////////////////////////////////////////////////////////////
    000542          T08S2:
828                 ;+
829                 ; FLOAT A ZERO ACROSS THE ID BUS DATA LINES
830                 ;-
831
832 000544                LOOP    I,1,32
833 000556                FLTZRO  T05L1,I         ; GENERATE THE TEST PATTERN
834 000564                ERLOOP
835 000566                INITIALIZE
836 000570                LOADCA  IDCS,CTMP7X     ; SET THE MAINTENANCE MODE BIT IN THE IDCS
837 000600                CLOCK   3               ; GO TO CPT150
838 000604                LOADCA  TOIDLO,T05L1    ; LOAD LOW 16 BITS
839 000614                LOADCA  TOIDHI,T05L1+2  ; AND UPPER 16 BITS
840 000624          SYNC10: CMPCAD  EQ,IDDATLO,T05L1 ; READ AND CHECK THE ID BUS DATA
841 000650                IFERROR 61,CIB6         ; BIT STUCK ON ID BUS OR CONSOLE ADAPTER
842 000656                ENDLOOP I               ; CONTINUE
843
844 000662                SUBTEST
                    ;////////////////////////////////////////////////////////////////////////////
    000662          T08S3:
845                 ;+
846                 ; NOW FLOAT A ONE THRU THE ID BUS DATA LINES
847                 ;-
848
849 000664                LOOP    I,1,32
850 000676                FLTONE  T05L1,I         ; GENERATE THE TEST PATTERN
851 000704                ERLOOP
852 000706                INITIALIZE
853 000710                LOADCA  IDCS,CTMP7X     ; SET MAINT MODE
854 000720                CLOCK   3
855 000724                LOADCA  TOIDLO,T05L1    ; LOAD LOW 16 BITS
856 000734                LOADCA  TOIDHI,T05L1+2  ; AND HIGH 16 BITS
857 000744          SYNC11: CMPCAD  EQ,IDDATLO,T05L1 ; READ AND CHECK THE ID BUS
858 000770                IFERROR 62,CIB6         ; BIT STUCK ON ID BUS OR CONSOLE BOARD
859 000776                ENDLOOP I               ; CONTINUE
860 001002                SKIP
861
862
863
864 001006  CIB6:     TSTVB   USCID             ; CHECK ID BUS ENABLE ON USC
865 001014            CHKPNT  CIB6A                     ; GO TO CIB6A IF OK
866 001022            REPORT  <USC>
867 001032  CIB6A:    TSTVB   ICLID                     ; CHECK ID BUS ENABLE ON ICL
868 001040            CHKPNT  CIB6B                     ; GO TO CIB6B IF OK
869 001046            REPORT  <ICL>
870 001056  CIB6B:    TSTVB   TBMID                     ; CHECK ID BUS ENABLE ON TBM
871 001064            CHKPNT  CIB6C                     ; GO TO CIB6C IF OK
872 001072            REPORT  <TBM>
873 001102  CIB6C:    TSTVB   IRCID                     ; CHECK VBUS ENABLE ON IRC
874 001110            CHKPNT  CIB6D                     ; GO TO CIB6D IF OK
875 001116            REPORT  <IRC,IDP>
876 001126  CIB6D:    TSTVB   SBHID                     ; CHECK ID BUS ENABLE ON SBH
877 001134            CHKPNT  CIB6E                     ; GO TO CIB6E IF OK
878 001142            REPORT  <SBH>
```

```
    879 001152  CIB6E:  REPORT  <CIB,IDBUS>
    880 001162          TESTDAT
                ;-------------------------------------------------------------------------------
    881 001166  CTMP7X: .WORD   IDMAINT+IDWRITE
    882 001170  T05L1:  .WORD   0,0
    883 001174  T05L2:  .WORD   125252,125252
    884 001200  T05L3:  .WORD   IDWRITE
    885 001202  USCID:  .WORD   1
    886 001204          VBUSG   0,20,1                  ; ID BUS XCVR EN L
    887 001206  ICLID:  .WORD   2
    888 001210          VBUSG   1,11,1                  ; EN ID XCEIV L
    889 001212          VBUSG   2,143,1                 ; D TO ID L
    890 001214  TBMID:  .WORD   1
    891 001216          VBUSG   3,2,1                   ; EN ID DRIVERS L
    892 001220  IRCID:  .WORD   1
    893 001222          VBUSG   3,127,1                 ; DATA EN L
    894 001224  SBHID:  .WORD   1
    895 001226          VBUSG   5,0,1                   ; EN ID DRIVERS L
    896 001230          ENDDAT
                ;-------------------------------------------------------------------------------
    897
```

```
    946         .SBTTL  T09     CONSOLE CLOCK CONTROL
                ;;*****************************************************************************
                ;++
                ;TEST   09      CONSOLE CLOCK CONTROL
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THAT THE CLOCK CYCLES THRU EACH TIME STATE
                ;       AND THAT A SINGLE BUS CYCLE LEAVES THE CLOCK IN CPT0. THIS IS
                ;       DONE BY SINGLE TIME STATING THE CLOCK AND CHECKING THE APPROPRIATE
                ;       VBUS SIGNAL AND BIT IN THE VBUS CONTROL REGISTER.
                ;
                ;       SINCE THE TIME STATES APPEAR ON THE V BUS ON MORE THAN ONE MODULE,
                ;       ALL THE V BUS BITS FOR THE PARTICULAR TIME STATE ARE EXAMINED. IF THE
                ;       FAILURE IS NOT THE FIRST CHECK OF THE TIME STATE, THE FAILURE IS MOST
                ;       LIKELY IN THE CLOCK TRANSLATOR ON THE MODULE THAT HAS THE V BUS BIT.
                ;
                ;       SUBTST 1 - CHECK TIME STATES 1, 2, 3, AND 4.
                ;       SUBTST 2 - CHECK THAT SINGLE BUS CYCLE STOPS THE CLOCK IN CPT0.
                ;                  THIS IS DONE BY SETTING THE CLOCK AT CPT1, EXECUTING A SINGLE
                ;                  BUS CYCLE, AND CHECKING THAT THE CLOCK IS IN CPT0.
                ;       SUBTST 3 - CHECK THAT THE PROCEED BIT STARTS THE CLOCK AND THAT THE
                ;                  CLOCK STOPPED BIT CLEARS.
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE CLOCK CONTROL LOGIC ON THE CIB AND CLK MODULES,
                ;       AND THE CLOCK TRANSLATORS ON SOME OF THE OTHER MODULES.
                ;
                ; ERROR DESCRIPTION
                ;       IF ERROR OCCURS AT SYNC19 OR SYNC 1B:
                ;               DATA: EXPECTED V BUS CHANNEL, BIT AND VALUE
                ;                     RECEIVED V BUS CHANNEL, BIT AND VALUE
                ;                     LOOP COUNT -- INDICATES WHICH TIME STATE SHOULD BE
                ;                                   ACTIVE, I.E. 1=CPT1, 2=CPT2, ETC.
                ;       IF ERROR OCCURS AT SYNC1A
                ;               DATA: EXPECTED CONTENTS OF V BUS CONTROL REGISTER
                ;                     RECEIVED CONTENTS OF V BUS CONTROL REGISTER
                ;                     LOOP COUNT -- SAME AS DESCRIBED ABOVE
                ;
                ;       IF ERROR OCCURS IN SUBTST 3:
                ;               DATA: EXPECTED CONTENTS OF MCR REG
                ;                     RECEIVED CONTENTS OF MCR REG
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC18--CLOCK IS SINGLE TIME STATED
                ;                  SYNC19--V BUS IS READ AND CHECKED
                ;                  SYNC1A--V BUS CONTROL REGISTER IS READ AND CHECKED
                ;       SUBTST 2 - SYNC1B--SINGLE BUS CYCLE IS EXECUTED
                ;       SUBTST 3 - SYNC99--PROCEED BIT IS SET
                ;                  SYNC9A--CLOCKED STOP BIT IS READ
                ;--
                ;*****************************************************************************
    001230      T09:
    950 001234          INITIALIZE
    951
    952 001236          SUBTEST
                ;/////////////////////////////////////////////////////////////////////////////
    001236      T09S1:
```

```
                        ;+
                        ; FIRST CHECK THAT SINGLE TIME STATE WORKS
                        ;-

001240                  ERLOOP
001242                  CLOCK   4               ; INITIALIZE THE STATE OF THE CLOCK (DOES AN SBC)
001246                  LOOP    I,1,4           ; TEST 4 TIME STATES
001260  SYNC18:         CLOCK   1               ; EXECUTE A SINGLE TIME STATE (STS)
001264  SYNC19:         TSTVB   CTMP10,I        ; LOOK FOR THE CORRECT CONDITIONS
001272                  IFERROR 63,CIB7         ; CORRECT TIME STATE DID NOT APPEAR
001300  SYNC1A:         CMPCAM  EQ,VBCTRL,CMSKX,,CTMP8X,I
001314                  IFERROR ,CIB7           ; VB CONTROL REG BIT'S BAD
001322                  ENDLOOP I               ; CONTINUE

001326                  SUBTEST
                        ;////////////////////////////////////////////////////////////////////////////////
001326  T09S2:
                        ;+
                        ; NOW CHECK THAT A SINGLE BUS CYCLE (SBC) TERMINATES IN CPT0
                        ;-

001330                  ERLOOP
001332                  INITIALIZE
001334                  CLOCK   1               ; PUT THE CLOCK IN TIME STATE 1
001340  SYNC1B:         CLOCK   4               ; EXECUTE A SINGLE BUS CYCLE
001344                  TSTVB   CTMP11          ; CHECK FOR CPT0
001352                  IFERROR 64,CIB7         ; SBC DID NOT TERMINATE IN CPT0

001360                  SUBTEST
                        ;////////////////////////////////////////////////////////////////////////////////
001360  T09S3:
                        ;+
                        ; NOW CHECK THAT THE CLOCK STOPPED BIT SETS AND CLEARS.
                        ;-

001362                  ERLOOP
001364  SYNC99:         LOADCA  CONMCR,T07L1    ; SET THE PROCEED BIT
001374                  CMPCAM  ,CONMCR,T07L2,,T07L4 ; CHECK THAT CLOCK STOPPED BIT CLEARED
001410                  IFERROR ,CIB7           ; CLOCK STOPPED BIT DID NOT CLEAR

                        ;+
                        ; NOW CHECK THAT THE BIT SETS
                        ;-

001416                  LOADCA  CONMCR,T07L3    ; SET THE SBC BIT
001426                  CMPCAM  ,CONMCR,T07L2,,T07L2 ; CHECK THAT CLOCK STOPPED BIT SET
001442                  IFERROR ,CIB7           ; CLOCK STOPPED BIT DID NOT SET
001450                  SKIP

001454  CIB7:           REPORT  <CLK,CIB>

001464                  TESTDAT
                        ;-------------------------------------------------------------------------------
001470  CTMP10:         .WORD   10
001472                  VBUSG   3,3,1           ; CPT0 L
001474                  VBUSG   4,26,0          ; CPT1 L
001476                  VBUSG   4,27,1          ; CPT1 H
```

```
1005 001500            VBUSG   4,106,0          ; CPT1 L
1006 001502            VBUSG   4,107,1          ; CPT1 H
1007 001504            VBUSG   4,25,0           ; CPT2 H
1008 001506            VBUSG   4,103,0          ; CPT2 H
1009 001510            VBUSG   4,24,0           ; CPT3
1010 001512            .WORD   10
1011 001514            VBUSG   3,3,1            ; CPT0
1012 001516            VBUSG   4,26,1           ; CPT1 L
1013 001520            VBUSG   4,27,0           ; CPT1 H
1014 001522            VBUSG   4,106,1          ; CPT1 L
1015 001524            VBUSG   4,107,0          ; CPT1 H
1016 001526            VBUSG   4,25,1           ; CPT2 H
1017 001530            VBUSG   4,103,1          ; CPT2 H
1018 001532            VBUSG   4,24,0           ; CPT3
1019 001534            .WORD   10
1020 001536            VBUSG   3,3,1            ; CPT0
1021 001540            VBUSG   4,26,1           ; CPT1 L
1022 001542            VBUSG   4,27,0           ; CPT1
1023 001544            VBUSG   4,106,1          ; CPT1 L
1024 001546            VBUSG   4,107,0          ; CPT1 H
1025 001550            VBUSG   4,25,0           ; CPT2 H
1026 001552            VBUSG   4,103,0          ; CPT2 H
1027 001554            VBUSG   4,24,1           ; CPT3
1028 001556  CTMP11:   .WORD   4
1029 001560            VBUSG   3,3,0            ; CPT0 L
1030 001562            VBUSG   4,27,0           ; CPT1
1031 001564            VBUSG   4,103,0          ; CPT2
1032 001566            VBUSG   4,24,0           ; CPT3
1033 001570  CTMP8X:   .WORD   CCPT1
1034 001572            .WORD   CCPT2
1035 001574            .WORD   CCPT3
1036 001576            .WORD   CCPT0
1037 001600  CMSKX:    .WORD   360
1038 001602  T07L1:    .WORD   CLRUWRD+PROCEED
1039 001604  T07L2:    .WORD   CLKSTPD
1040 001606  T07L3:    .WORD   CLRUWRD+SBC
1041 001610  T07L4:    .WORD   0
1042 001612            ENDDAT
             ;-------------------------------------------------------------------------------

1043
```

```
                        .SBTTL  TOA     CONSOLE ID CYCLE FUNCTION
                        ;;*****************************************************************************
                        ;++
                        ;TEST   0A      CONSOLE ID CYCLE FUNCTION
                        ;
                        ; TEST DESCRIPTION
                        ;       THIS TEST CHECKS THE ID CYCLE FUNCTION. THIS IS DONE BY SETTING
                        ;       ID CYCLE, EXECUTING A SINGLE BUS CYCLE, CHECKING TO SEE IF ID
                        ;       MAINTENANCE SET, EXECUTING ANOTHER BUS CYCLE, AND CHECKING THAT
                        ;       ID MAINTENANCE AND CYCLE CLEARED.
                        ;
                        ; LOGIC DESCRIPTION
                        ;       THIS TEST CHECKS THE ID CYCLE AND ID MAINTENANCE FLIP FLOPS ON
                        ;       THE CIB MODULE.
                        ;
                        ; ERROR DESCRIPTION
                        ;       DATA: EXPECTED CONTENTS OF THE IDCS REGISTER
                        ;             RECEIVED CONTENTS OF THE IDCS REGISTER
                        ;
                        ; SYNC POINT DESCRIPTION
                        ;       ADDRESS SYNC1C--ID MAINTENANCE BIT SHOULD SET
                        ;       ADDRESS SYNC1D--ID CYCLE AND ID MAINTENANCE SHOULD CLEAR.
                        ;--
                        ;;*****************************************************************************
     001612             TOA:
1069 001616                     INITIALIZE
1070
1071                    ;+
1072                    ; FIRST CHECK THAT ID MAINTENANCE SETS WITH CPTO
1073                    ;-
1074
1075 001620                     ERLOOP
1076 001622                     LOADCA  IDCS,CTMP12     ; SET THE CYCLE BIT
1077 001632             SYNC1C: CLOCK   4               ; EXECUTE A SINGLE BUS CYCLE
1078 001636                     CMPCAM  EQ,IDCS,T08M1,,CTMP12+2 ; CHECK FOR ID MAINT AND CYCLE SET
1079 001652                     IFERROR 65,CIB10        ; ID MAINTENANCE DID NOT SET AFTER ID CYCLE
1080
1081                    ;+
1082                    ; NOW CHECK THAT ID MAINTENANCE AND CYCLE CLEARS ON THE NEXT CPTO
1083                    ;-
1084
1085 001660             SYNC1D: CLOCK   4               ; EXECUTE ANOTHER BUS CYCLE
1086 001664                     CMPCAM  EQ,IDCS,T08M1,,T08L1 ; CHECK THAT ID MAINTENANCE  AND CYCLE CLEARED
1087 001700                     IFERROR 66,CIB10        ; ID MAINTENANCE DID NOT CLEAR WHEN CYCLE COMPLETED
1088 001706                     SKIP
1089
1090 001712             CIB10:  REPORT  <CIB>
1091
1092 001722                     TESTDAT
                        ;-------------------------------------------------------------------------------
1093 001726             CTMP12: .WORD   IDCYCLE         ; ID CYCLE BIT
1094 001730                     .WORD   IDCYCLE+IDMAINT
1095 001732             T08L1:  .WORD   0
1096 001734             T08M1:  .WORD   100377
1097                    ;    HEXDATA    80FF
1098 001736                     ENDDAT
                        ;-------------------------------------------------------------------------------
```

```
   1099
   1134
```

```
1138                .SBTTL  TOB     CONSL FROM ID REG CLK CTRL & DATA INTEG
                    ;;*****************************************************************
                    ;++
                    ;TEST   0B      CONSL FROM ID REG CLK CTRL & DATA INTEG
                    ;
                    ; TEST DESCRIPTION
                    ;       THIS TEST FLOATS A ZERO AND A ONE THRU THE "FROM ID" REGISTER
                    ;       BY LOADING THE "TO ID" REGISTER WITH THE DATA PATTERN, AND SELECTING
                    ;       THE ID BUS ADDRESS FOR THE "FROM ID" REGISTER IN THE IDCS REGISTER,
                    ;       EXECUTING A SINGLE BUS CYCLE, AND CHECKING THAT THE DATA WAS WRITTEN
                    ;       INTO THE "FROM ID" REGISTER.
                    ;
                    ;       THEN THE FROM ID REGISTER IS LOADED WITH THE ID DATA BY EXECUTING
                    ;       AN ID BUS READ FUNCTION WITH THE "TO ID" REGISTER AS THE REGISTER
                    ;       BEING READ.
                    ;
                    ;       SUBTST 1 - FLOAT A ZERO THRU THE "FROM ID" WITH ID BUS WRITES.
                    ;       SUBTST 2 - FLOAT A ONE THRU THE "FROM ID" REG WITH ID BUS WRITES.
                    ;       SUBTST 3 - CHECK THAT THE "FROM ID" REGISTER IS LOADED WHEN AN ID
                    ;                  BUS READ IS EXECUTED, IN MAINTENANCE MODE.
                    ;
                    ; LOGIC DESCRIPTION
                    ;       THIS TEST CHECKS THE "FROM ID" REGISTER CLOCK CONTROL ON THE
                    ;       CIB MODULE AND THE Q BUS INTERFACE TO THE "FROM ID" REGISTER.
                    ;
                    ; ERROR DESCRIPTION
                    ;       DATA: EXPECTED CONTENTS OF THE "FROM ID" REGISTER
                    ;             RECEIVED CONTENTS OF THE "FROM ID" REGISTER
                    ;             LOOP COUNT -- INDICATES THE BIT POSITION (OR DATA PATTERN FOR
                    ;               SUBTEST 3) THAT IS CURRENTLY BEING TESTED, I.E. 1=BIT 0
                    ;               2=BIT 1, 3=BIT 2, 4=BIT 3, ETC.
                    ;
                    ; SYNC POINT DESCRIPTION
                    ;       SUBTST 1 - SYNC1E--LOAD THE "FROM ID" REG WITH A WRITE FUNCTION
                    ;                  SYNC1F--READ THE FROM ID REGISTER OVER THE Q BUS
                    ;       SUBTST 2 - SYNC20--LOAD THE FROM ID REG WITH A WRITE FUNCTION
                    ;                  SYNC21--READ THE FROM ID REG OVER THE Q BUS
                    ;       SUBTST 3 - SYNC22--LOAD THE FROM ID REG WITH AN ID BUS READ FUNCTION
                    ;--
                    ;;*****************************************************************
    001736          T0B:
1142 001742                 INITIALIZE
1143
1144 001744                 SUBTEST
                    ;/////////////////////////////////////////////////////////////////
    001744          T0BS1:
1145                ;+
1146                ; FIRST FLOAT A ZERO THRU THE FROM ID REG WITH ID BUS WRITES.
1147                ;-
1148
1149 001746                 LOOP    I,1,32
1150 001760                 FLTZRO  T09L1,I         ; GENERATE TEST PATTERN
1151 001766                 ERLOOP
1152 001770     SYNC1E:     LDIDREG CONTXD,T09L1    ; WRITE THE FROM ID REG OVER THE ID BUS
1153 001776     SYNC1F:     CMPCAD  EQ,FMIDLO,T09L1 ; READ AND CHECK THE FROM ID REG
1154 002022                 IFERROR 67,CIB11        ; FROM ID REG DID NOT LOAD THE ID DATA
1155 002030                 ENDLOOP I
```

```
                        SKIP    SUB

                CIB11:  REPORT  <CIB,IDBUS>

                        SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
                TOBS2:
                ;+
                ; NOW FLOAT A ONE THRU THE FROM ID REGISTER WITH ID BUS WRITES
                ;-

                        LOOP    I,1,32
                        FLTONE  T09L1,I
                        ERLOOP
                SYNC20: LDIDREG CONTXD,T09L1    ; LOAD THE TEST PATTERN
                SYNC21: CMPCAD  EQ,FMIDLO,T09L1 ; CHECK THE REGISTER
                        IFERROR ,CIB11          ; FROM ID REG FAILED
                        ENDLOOP I               ; CONTINUE

                        SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
                TOBS3:
                ;+
                ; THIS SUBTEST CHECKS THAT THE FROM ID REGISTER GETS LOADED WHEN
                ; AN ID BUS READ FUNCTION IS ISSUED.
                ;-

                        LOOP    I,1,2
                        ERLOOP
                        LOADCA  IDCS,CX1TMP     ; SETUP TO DO THE READ
                        LOADCA  TOIDLO,CX2TMP,I ; LOAD DATA TO READ
                        LOADCA  TOIDHI,CX3TMP,I ; ...
                SYNC22: CLOCK   4               ; READ THE TO ID REG INTO THE FROM ID REG
                        CMPCAD  EQ,FMIDLO,CTMP13,I ; CHECK THE RESULT
                        IFERROR ,CIB11
                        ENDLOOP I               ; CONTINUE

                        TESTDAT
                ;-------------------------------------------------------------------------------
                CTMP13: .WORD   125252,125252
                ;   HEXDATA      AAAA, AAAA
                CTMP14: .WORD   52525,52525
                ;   HEXDATA      5555, 5555
                CX1TMP: .WORD   IDMAINT+CONRXD
                CX2TMP: .WORD   125252,52525
                CX3TMP: .WORD   125252,52525
                T09L1:  .WORD   0,0
                        ENDDAT
                ;-------------------------------------------------------------------------------
```

```
1234                    .SBTTL  SECTION NUMBER 03
     000000                     .PSECT  OVR03
                        .SBTTL  TOC     CONSOLE MAINTENANCE RETURN
                        ;;*****************************************************************************
                        ;++
                        ;TEST   0C      CONSOLE MAINTENANCE RETURN
                        ;
                        ; TEST DESCRIPTION
                        ;       THIS TEST ENSURES THAT MAINTENANCE RETURN SETS AT CPT150
                        ;       AND CLEARS AT CPT50 AND THAT MAINTENANCE RETURN IS LATCHED ON THE MICRO SEQ.
                        ;       THIS IS DONE BY SETTING THE MAINTENANCE RETURN BIT IN THE MACHINE
                        ;       CONTROL REGISTER, TICKING THE CLOCK TO CPT150, EXAMINING THE MAINTENANCE
                        ;       RETURN SIGNAL GOING TO THE MICRO SEQUENCER FROM THE CONSOLE ON THE
                        ;       V BUS, TICKING THE CLOCK TO CPT0 AND CHECKING THAT THE MAINTENANCE
                        ;       RETURN BIT IN THE MCR REG CLEARED, AND THAT THE MAINTENANCE RETURN
                        ;       LATCH ON THE MICRO SEQUENCER SET, TICKING THE CLOCK TO CPT50 AND
                        ;       CHECKING THAT THE MAINTENANCE RETURN SIGNAL GOING TO THE MICRO
                        ;       SEQUENCER CLEARED. THEN TICKING THE CLOCK TO CPT0 AND CHECKING
                        ;       THAT THE MAINT RET LATCH CLEARED ON THE MICRO SEQ.
                        ;
                        ; LOGIC DESCRIPTION
                        ;       THIS TEST CHECKS THE TWO MAINTENANCE RETURN FLIP FLOPS ON THE CIB
                        ;       MODULE AND THE MAINTENANCE RETURN LATCH ON THE MICRO SEQUENCER
                        ;       MODULE.
                        ;
                        ; ERROR DESCRIPTION
                        ;       DATA: EXPECTED V BUS CHANNEL, BIT AND VALUE (OR BIT 10 OF THE MCR REG)
                        ;             RECEIVED V BUS CHANNEL, BIT AND VALUE (OR BIT 10 OF THE MCR REG)
                        ;
                        ; SYNC POINT DESCRIPTION
                        ;       ADDRESS SYNC23--READ AND CHECK THE MNT RTN LATCH ON THE MICRO SEQ.
                        ;       ADDRESS SYNC24--''D MAINT RET'' SETS ON THE CIB MODULE
                        ;       ADDRESS SYNC25--MAINT RET CLEARS IN THE MCR REG AND LATCHES IN THE
                        ;                       MICRO SEQUENCER.
                        ;       ADDRESS SYNC26--''D MAINT RET'' CLEARS ON THE CONSOLE
                        ;       ADDRESS SYNC27--THE MAINT RET LATCH CLEARS ON THE MICRO SEQUENCER.
                        ;--
                        ;;*****************************************************************************
     000034             TOC:
1238 000040                     ERLOOP
1239 000042                     INITIALIZE
1240 000044             SYNC23: TSTVB   VBMNT4          ; CHECK THAT THE MNT RTN LATCH IS CLEAR
1241                                                    ; ON THE MICRO SEQ.
1242 000052                     IFERROR ,CIB12A         ; INIT DID NOT CLEAR MNT RET LATCH ON SEQ
1243
1244 000060                     ERLOOP
1245 000062                     INITIALIZE
1246 000064                     LOADCA  CONMCR,CTMP99   ; SET THE MAINTENANCE RETURN BIT
1247 000074                     CLOCK   1               ; GO TO TIME STATE 50
1248 000100                     CLOCK   1               ; GO TO TIME STATE 100
1249 000104             SYNC24: CLOCK   1               ; GO TO TIME STATE 150
1250 000110                     TSTVB   VBMNT1          ; CHECK THAT D MAINTENANCE RETURN SET
1251 000116                     IFERROR 70,CIB12        ; CONSOLE MAINTENANCE RETURN DID NOT SET
1252
1253 000124             SYNC25: CLOCK   1               ; STEP TO CPT0
1254 000130                     TSTVB   VBMNT2          ; CHECK THAT MAINT RET SET ON THE SEQ.
1255 000136                     IFERROR 71,CIB12A       ; THE LATCH ON THE SEQ DID NOT SET
```

```
1256
1257 000144          CMPCAM  ,CONMCR,CTMP96,,TOAL2 ; CHECK THAT BIT 10 IN MCR CLEARED
1258 000160          IFERROR ,CIB12           ; BIT 10 IN MCR REG DID NOT CLEAR
1259
1260 000166  SYNC26: CLOCK   1               ; GO TO CPT50
1261 000172          TSTVB   VBMNT3          ; CHECK THAT D MAINT RET CLEARED
1262 000200          IFERROR ,CIB12           ; D MAINT RET DID NOT CLEAR
1263
1264 000206          CLOCK   1               ; GO TO CPT100
1265 000212          CLOCK   1               ; GO TO CPT150
1266 000216  SYNC27: CLOCK   1               ; GO TO CPT0
1267 000222          TSTVB   VBMNT4          ; CHECK THAT MNT RET LATCH CLEARED ON SEQ
1268 000230          IFERROR ,CIB12A          ; MNT RET LATCH DID NOT CLEAR ON SEQ.
1269 000236          SKIP
1270
1271 000242  CIB12:  REPORT  <CIB>
1272 000252  CIB12A: REPORT  <USC>
1273 000262          TESTDAT
                     ;-------------------------------------------------------------------------------
1274 000266  CTMP99: .WORD   MNTRTN+SBC+STS
1275 000270  CTMP97: .WORD   MNTRTN+STS+SBC+PROCEED
1276 000272  CTMP96: .WORD   MNTRTN
1277 000274  VBMNT1: .WORD   1
1278 000276          VBUSG   0,117,1         ; D MAINT RET H
1279 000300  VBMNT2: .WORD   1
1280 000302          VBUSG   0,124,1         ; MAINT RET (1) H
1281 000304  VBMNT3: .WORD   1
1282 000306          VBUSG   0,117,0         ; D MAINT RET H
1283 000310  VBMNT4: .WORD   1
1284 000312          VBUSG   0,124,0         ; MAINT RET (1) H
1285 000314  TOAL2:  .WORD   0
1286 000316          ENDDAT
                     ;-------------------------------------------------------------------------------
1287
```

```
1328            .SBTTL  TOD     RXCS REGISTER FROM THE ID BUS SIDE
                ;;*****************************************************************************
                ;++
                ;TEST   OD      RXCS REGISTER FROM THE ID BUS SIDE
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST ENSURES THAT THE INTERRUPT ENABLE (IE) BIT IN THE
                ;       RXCS REGISTER CAN BE SET AND CLEARED FROM THE ID BUS, AND
                ;       THAT THE DONE BIT CAN BE READ.
                ;       IT THEN CHECKS THAT THE DONE BIT CLEARS WHEN THE RX DATA REGISTER IS
                ;       READ FROM THE ID BUS.
                ;       THEN A TEST IS MADE TO ENSURE THAT AN LSI-11 RESET CLEARS THE INTERRUPT
                ;       ENABLE BIT.
                ;
                ;       SUBTST 1 - CHECK THAT THE WRITES ON THE ID BUS WILL SET AND CLEAR
                ;                  THE RECEIVER INTERRUPT ENABLE BIT.
                ;       SUBTST 2 - CHECK THAT THE DONE BIT IS GATED ONTO THE ID BUS WHEN
                ;                  THE RXCS IS READ.
                ;       SUBTST 3 - CHECK THAT THE DONE BIT CLEARS WHEN THE RECEIVER
                ;                  DATA BUFFER IS READ ON THE ID BUS.
                ;       SUBTST 4 - CHECK THAT AN LSI-11 RESET CLEARS THE IE BIT.
                ;       SUBTST 5 - CHECK THAT A STAR INIT CLEARS THE IE BIT.
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE ID BUS INTERFACE TO THE RECEIVER INTERRUPT
                ;       ENABLE AND DONE BITS AND THE DONE BIT CLEAR CONTROL ON THE
                ;       CIB MODULE.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED CONTENTS OF THE RECEIVER CONTROL AND STATUS REG
                ;             RECEIVED CONTENTS OF THE RECEIVER CONTROL AND STATUS REG
                ;             LOOP COUNT - SUBTST 1 ONLY -- INDICATES WHICH DATA PATTERN IS BEING
                ;               USED, I.E. 1=40(X), 2=00
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC28--IE BIT IS WRITTEN
                ;                  SYNC29--IE BIT IS READ OVER THE ID BUS
                ;       SUBTST 2 - SYNC2A--DONE BIT IS READ OVER THE ID BUS
                ;       SUBTST 3 - SYNC2B--DONE BIT SHOULD CLEAR
                ;       SUBTST 4 - SYNC9C--IE BIT SHOULD CLEAR
                ;       SUBTST 5 - SYNCA0--IE BIT SHOULD CLEAR
                ;--
                ;;*****************************************************************************
    000316      TOD:
1332 000322             INITIALIZE
1333
1334 000324             SUBTEST
                ;//////////////////////////////////////////////////////////////////////////////
    000324      TODS1:
1335            ;+
1336            ; CHECK THAT THE ID BUS CAN SET AND CLEAR THE INTERRUPT ENABLE BIT.
1337            ;-
1338
1339 000326             LOADCA  RXDNE,TOBL1     ; ENSURE THAT DONE IS CLEAR
1340 000336             LOOP    I,1,2
1341 000350             ERLOOP
1342 000352     SYNC28: LDIDREG CONRXS,CTMP17,I ; SET (OR CLEAR) THE IE BIT
```

```
1343 000360  SYNC29: READID  CONRXS            ; READ THE IE BIT
1344 000364          CMPCAD  EQ,IDREGLO,CTMP17,I ; CHECK THAT THE BIT IS OK
1345 000410          IFERROR 72,CIB13          ; RX IE BIT DID NOT SET OR CLEAR
1346 000416          ENDLOOP I                 ; CONTINUE
1347
1348 000422          SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
      000422    TODS2:
1349              ;+
1350              ; NOW CHECK THAT THE DONE BIT CAN BE READ AS A ONE
1351              ;-
1352
1353 000424          LOADCA  RXDNE,TOBL1+2     ; SET THE DONE BIT
1354 000434          ERLOOP
1355 000436  SYNC2A: READID  CONRXS            ; READ THE DONE BIT
1356 000442          CMPCAD  EQ,IDREGLO,TOBL1+2 ; CHECK THAT DONE CAN BE READ AS A ONE
1357 000466          IFERROR 100,CIB13         ; RXDNE BIT DID NOT GET GATED ONTO ID BUS
1358
1359 000474          SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
      000474    TODS3:
1360              ;+
1361              ; NOW CHECK THAT THE RX DONE BIT CLEARS WHEN THE RX DATA BUFFER IS
1362              ; READ FROM THE ID BUS.
1363              ;-
1364
1365 000476          ERLOOP
1366 000500          LOADCA  RXDNE,TOBL1+2     ; SET THE RXDNE BIT
1367 000510          LOADCA  IDCS,CTMP20       ; SET THE ID BUS TO DO A READ
1368 000520          CLOCK   1                 ; GO TO CPT50
1369 000524          CLOCK   1                 ; GO TO CPT100
1370 000530          CLOCK   1                 ; GO TO CPT150
1371 000534  SYNC2B: CLOCK   1                 ; GO TO CPT0
1372 000540          CMPCA   EQ,RXDNE,CTMP17+2 ; CHECK THAT RX DONE WAS CLEARED
1373 000552          IFERROR 73,CIB13          ; RXDNE DID NOT CLEAR WHEN THE RX DATA
1374                                           ; DATA BUFFER WAS READ
1375
1376 000560          SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
      000560    TODS4:
1377              ;+
1378              ; NOW CHECK THAT AN LSI-11 RESET CLEARS THE IE BIT
1379              ;-
1380
1381 000562          ERLOOP
1382 000564          LDIDREG CONRXS,CTMP17     ; SET THE IE BIT
1383 000572  SYNC9C: RESETC                    ; EXECUTE THE RESET
1384 000574          READID  CONRXS            ; READ THE IE BIT
1385 000600          CMPCA   EQ,IDREGLO,CTMP17+2 ; CHECK THAT THE IE BIT CLEARED
1386 000612          IFERROR ,CIB13            ; RESET DID NOT CLEAR IE BIT
1387
1388 000620          SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
      000620    TODS5:
1389              ;+
1390              ; CHECK THAT A STAR RESET CLEARS THE IE BIT
1391              ;-
```

```
1392
1393 000622          ERLOOP
1394 000624          LDIDREG CONRXS,CTMP17   ; SET THE IE BIT
1395 000632  SYNCA0: LOADCA  CONMCR,TOBL2    ; SET THE SYSTEM INIT BIT
1396 000642          LOADCA  CONMCR,TOBL3    ; CLEAR SYSTEM INIT
1397 000652          READID  CONRXS          ; READ THE IE BIT
1398 000656          CMPCA   EQ,IDREGLO,CTMP17+2 ; CHECK THAT IE BIT CLEARED
1399 000670          IFERROR ,CIB13          ; IE BIT DID NOT CLEAR WITH A SYSTEM INIT
1400 000676          SKIP
1401
1402 000702  CIB13:  REPORT  <CIB>
1403
1404 000712          TESTDAT
                     ;-------------------------------------------------------------------------------
1405 000716  CTMP17: .WORD   100,0            ; INTERRUPT ENABLE
1406 000722          .WORD   0,0
1407 000726  CTMP20: .WORD   IDMAINT+CONRXD
1408 000730  TOBL1:  .WORD   0,200,0
1409 000736  TOBL2:  .WORD   INIT+CLRUWRD+SBC
1410 000740  TOBL3:  .WORD   CLRUWRD+SBC
1411 000742          ENDDAT
                     ;-------------------------------------------------------------------------------
1412
```

```
1448            .SBTTL  TOE     TXCS REGISTER ON THE ID BUS
                ;;*****************************************************************************
                ;++
                ;TEST   OE      TXCS REGISTER ON THE ID BUS
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THAT THE TRANSMITTER INTERRUPT ENABLE (TXIE) BIT IN THE
                ;       TRANSMITTER STATUS (TXCS) REGISTER CAN BE SET AND CLEARED FROM
                ;       THE ID BUS, AND THAT THE READY BIT CAN BE READ.
                ;       IT THEN CHECKS THAT THE TRANSMITTER READY BIT IS
                ;       CLEARED WHEN THE TRANSMITTER DATA BUFFER IS WRITTEN INTO.
                ;       IT THEN CHECKS THAT THE IE BIT CLEARS WITH AN LSI-11 RESET.
                ;
                ;       SUBTST 1 - CHECK THAT THE IE BIT CAN BE SET AND CLEARED
                ;       SUBTST 2 - CHECK THAT THE READY BIT CAN BE READ AS A ONE
                ;       SUBTST 3 - CHECK THAT THE READY BIT CLEARS WHEN THE ID BUS WRITES
                ;                  TO THE TRANSMITTER DATA BUFFER.
                ;       SUBTST 4 - CHECK THAT THE IE BIT CLEARS WITH AN LSI-11 RESET.
                ;       SUBTST 5 - CHECK THAT THE IE BIT CLEARS WITH A STAR INIT
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE ID BUS INTERFACE TO THE TRANSMITTER CONTROL
                ;       AND STATUS REGISTER AND THE CLEAR CONTROL ON THE TRANSMITTER
                ;       READY BIT ON THE CIB MODULE.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED CONTENTS OF THE TXCS REGISTER
                ;             RECEIVED CONTENTS OF THE TXCS REGISTER
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC2C--TRANSMITTER IE BIT IS WRITTEN
                ;                  SYNC2D--TRANSMITTER IE BIT IS READ OVER THE ID BUS
                ;       SUBTST 2 - SYNC2E--TRANSMITTER READY BIT IS READ OVER THE ID BUS
                ;       SUBTST 3 - SYNC2F--TRANSMITTER READY BIT SHOULD CLEAR
                ;       SUBTST 4 - SYNC9D--IE BIT SHOULD CLEAR
                ;       SUBTST 5 - SYNCA1--IE BIT CHOULD CLEAR
                ;--
                ;;*****************************************************************************
    000742      TOE:
1452 000746             INITIALIZE
1453 000750             LOADCA  TXRDY,TOCL1     ; ENSURE READY BIT IS CLEAR
1454
1455 000760             SUBTEST
                ;//////////////////////////////////////////////////////////////////////////////
    000760      TOES1:
1456            ;+
1457            ; CHECK THAT THE IE BIT CAN BE WRITTEN AS A ONE AND A ZERO
1458            ;-
1459
1460 000762             LOOP    I,1,2
1461 000774             ERLOOP
1462 000776     SYNC2C: LDIDREG CONTXS,TOCL3,I  ; SET (OR CLEAR) THE IE BIT
1463 001004     SYNC2D: READID  CONTXS          ; GET THE BIT BACK
1464 001010             CMPCAD  EQ,IDREGLO,TOCL3,I ; CHECK THAT BIT IS CORRECT
1465 001034             IFERROR 75,CIB14        ; TRANSMITTER IE DID NOT SET OR CLEAR
1466 001042             ENDLOOP I               ; CONTINUE
1467
```

```
                        SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
        001046  TOES2:
                ;+
                ; NOW CHECK THAT THE READY BIT CAN BE READ AS A ONE
                ;-

        001050          ERLOOP
        001052          LOADCA  TXRDY,TOCL1+2   ; SET THE READY BIT
        001062  SYNC2E: READID  CONTXS          ; READ THE TXCS REG
        001066          CMPCAD  EQ,IDREGLO,TOCL1+2 ; CHECK THAT READY BIT WAS READ AS A ONE
        001112          IFERROR 101,CIB14       ; TXRDY DID NOT GET GATED ONTO ID BUS

        001120          SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
        001120  TOES3:
                ;+
                ; NOW CHECK THAT THE TRANSMITTER READY BIT CLEARS WHEN THE TRANSMITTER
                ; DATA BUFFER IS WRITTEN INTO.
                ;-

        001122          ERLOOP
        001124          LOADCA  TXRDY,TOCL1+2   ; SET THE TRANSMITTER READY BIT
        001134          LOADCA  IDCS,CTMP16     ; SET ID CONTROL TO WRITE TXDB
        001144          CLOCK   1               ; GO TO CPT50
        001150          CLOCK   1               ; GO TO CPT100
        001154          CLOCK   1               ; GO TO CPT150
        001160  SYNC2F: CLOCK   1               ; GO TO CPT0
        001164          CMPCA   EQ,TXRDY,TOCL3+2 ; CHECK THAT READY BIT CLEARED
        001176          IFERROR 76,CIB14        ; TRANSMITTER READY BIT DID NOT CLEAR

        001204          SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
        001204  TOES4:
                ;+
                ; NOW CHECK THAT AN LSI-11 RESET CLEARS THE IE BIT
                ;-

        001206          ERLOOP
        001210          LDIDREG CONTXS,TOCL3    ; SET THE IE BIT
        001216  SYNC9D: RESETC                  ; EXECUTE THE RESET
        001220          READID  CONTXS          ; READ THE IE BIT
        001224          CMPCA   EQ,IDREGLO,TOCL3+2 ; CHECK THAT IT CLEARED
        001236          IFERROR ,CIB14          ; IE BIT DID NOT CLEAR

        001244          SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
        001244  TOES5:
                ;+
                ; CHECK THAT THE IE BIT CLEARS WITH A STAR INIT
                ;-

        001246          ERLOOP
        001250          LDIDREG CONTXS,TOCL3    ; SET THE IE BIT
        001256  SYNCA1: LOADCA  CONMCR,TOCL4    ; SET THE INIT BIT
        001266          LOADCA  CONMCR,TOCL5    ; CLEAR THE INIT BIT
        001276          READID  CONTXS          ; READ THE IE BIT
```

TOE TXCS REGISTER ON THE ID BUS

```
1517 001302          CMPCA   EQ,IDREGLO,TOCL3+2 ; CHECK THAT IT CLEARED
1518 001314          IFERROR ,CIB14          ; IE BIT DID NOT CLEAR WITH STAR INIT
1519 001322          SKIP
1520
1521 001326  CIB14:  REPORT  <CIB>
1522
1523 001336          TESTDAT
                     ;-------------------------------------------------------------------------
1524 001342  CTMP16: .WORD   IDMAINT+IDWRITE+CONTXD
1525 001344  TOCL1:  .WORD   0,200,0
1526 001352  TOCL2:  .WORD   125252,125252
1527         ;   HEXDATA    AAAA,  AAAA
1528 001356  TOCL3:  .WORD   100,0,0,0
1529 001366  TOCL4:  .WORD   INIT+CLRUWRD+SBC
1530 001370  TOCL5:  .WORD   CLRUWRD+SBC
1531 001372          ENDDAT
                     ;-------------------------------------------------------------------------
1532
```

```
1559            .SBTTL  TOF     ID BUS REGISTER ADDRESS INTEGRITY
                ;;******************************************************************************
                ;++
                ;TEST   OF      ID BUS REGISTER ADDRESS INTEGRITY
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST DOES A CURSORY CHECK OF THE ID BUS REGISTER
                ;       DUAL ADDRESSING. THIS IS DONE BY WRITING A DIFFERENT PATTERN
                ;       IN 28 OF THE REGISTERS AND THEN CHECKING THE CONTENTS OF THE
                ;       REGISTERS. WHEN THE REGISTER IS WRITTEN, IT IS CHECKED TO ENSURE THE DATA
                ;       WAS PUT IN THE REGISTER.
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST WILL DETECT ERRORS IN ID BUS REGISTER DECODE LOGIC
                ;       OR IN THE REGISTERS THEMSELVES. ONLY ONE OR TWO BITS ARE SET
                ;       IN EACH REGISTER TO MINIMIZE THE PROBABILITY OF A STUCK
                ;       BIT IN THE REGISTER SHOWING UP.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED CONTENTS OF THE ID BUS REGISTER
                ;             RECEIVED CONTENTS OF THE ID BUS REGISTER
                ;             LOOP COUNT -- INDICATES WHICH REGISTER DID NOT HAVE THE
                ;               CORRECT DATA, I.E. 1=USCBRK, 2=USCADR, 3=POBR, ETC.
                ;
                ; SYNC POINT DESCRIPTION
                ;       ADDRESS SYNC30--ID BUS REGISTER IS WRITTEN
                ;       ADDRESS SYNC31--ID BUS REGISTER IS READ
                ;--
                ;;******************************************************************************
    001372      TOF:
1563 001376             INITIALIZE
1564
1565            ;+
1566            ; LOAD THE REGISTERS WITH DIFFERANT DATA PATTERNS
1567            ;-
1568
1569 001400             LOOP    I,1,28
1570 001412             ERLOOP
1571 001414             LOADCA  IDCS,TMP510,I   ; LOAD THE REGISTER ADDRESS
1572 001424             LOADCA  TOIDLO,TMP511,I ; LOAD THE DATA PATTERN
1573 001434     SYNC30: CLOCK   4               ; WRITE THE REGISTER
1574 001440             LOADCA  IDCS,TMP512,I   ; LOAD IDCS TO READ THE REGISTER
1575 001450             CLOCK   3               ; GO TO CPT3 TO READ THE REGISTER
1576 001454             CMPCA   ,IDDATLO,TMP511,I ; CHECK THAT THE DATA WAS WRITTEN
1577 001466             CLOCK   1               ; GO BACK TO CPTO
1578 001472             IFERROR ,ID1            ; REGISTER DID NOT WRITE CORRECTLY
1579 001500             ENDLOOP I               ; CONTINUE
1580
1581            ;+
1582            ; NOW CHECK THE CONTENTS OF THE REGISTERS
1583            ;-
1584
1585 001504             LOOP    I,1,28
1586 001516             ERLOOP
1587 001520             LOADCA  IDCS,TMP512,I   ; LOAD THE REGISTER ADDRESS
1588 001530             CLOCK   2               ; ADVANCE TO CPT2
1589 001534     SYNC31: CMPCA   EQ,IDDATLO,TMP511,I ; CHECK THE DATA ON THE ID BUS
```

```
                        CLOCK   2
                        IFERROR ,ID1            ; ID BUS DUAL ADDRESSING PROBLEM
                        ENDLOOP I               ; CONTINUE

                ;+
                ; NOW CLEAN UP THE REGISTERS
                ;-

                        LOOP    I,1,28
                        LOADCA  IDCS,TMP510
                        LOADCA  TOIDLO,TMP513
                        LOADCA  TOIDHI,TMP513
                        CLOCK   4
                        ENDLOOP I
                        SKIP

ID1:                    REPORT  <IDBUS>

                        TESTDAT
                ;-------------------------------------------------------------------------------
TMP513:         .WORD   0
TMP510:         .WORD   IDMAINT+IDWRITE+USCBRK
                .WORD   IDMAINT+IDWRITE+USCADR
                .WORD   IDMAINT+IDWRITE+P0BR
                .WORD   IDMAINT+IDWRITE+P1BR
                .WORD   IDMAINT+IDWRITE+SBR
                .WORD   IDMAINT+IDWRITE+KSP
                .WORD   IDMAINT+IDWRITE+ESP
                .WORD   IDMAINT+IDWRITE+SSP
                .WORD   IDMAINT+IDWRITE+USP
                .WORD   IDMAINT+IDWRITE+ISP
                .WORD   IDMAINT+IDWRITE+FPDA
                .WORD   IDMAINT+IDWRITE+D.SV
                .WORD   IDMAINT+IDWRITE+Q.SV
                .WORD   IDMAINT+IDWRITE+TEMP0
                .WORD   IDMAINT+IDWRITE+TEMP1
                .WORD   IDMAINT+IDWRITE+TEMP2
                .WORD   IDMAINT+IDWRITE+TEMP3
                .WORD   IDMAINT+IDWRITE+TEMP4
                .WORD   IDMAINT+IDWRITE+TEMP5
                .WORD   IDMAINT+IDWRITE+TEMP6
                .WORD   IDMAINT+IDWRITE+TEMP7
                .WORD   IDMAINT+IDWRITE+TEMP8
                .WORD   IDMAINT+IDWRITE+TEMP9
                .WORD   IDMAINT+IDWRITE+PCBB
                .WORD   IDMAINT+IDWRITE+SCBB
                .WORD   IDMAINT+IDWRITE+P0LR
                .WORD   IDMAINT+IDWRITE+P1LR
                .WORD   IDMAINT+IDWRITE+SLR
TMP511:         .WORD   401
                .WORD   402
                .WORD   404
                .WORD   410
                .WORD   420
                .WORD   1000,1001,1002,1004,1010,1020,1040,1100,1200,1400
                .WORD   2000,2001,2002,2004,2010,2020,2040,2100,2200,2400
                .WORD   4000,4001,4002
```

TOF ID BUS REGISTER ADDRESS INTEGRITY

```
1646 002040  TMP512: .WORD   IDMAINT+USCBRK
1647 002042          .WORD   IDMAINT+USCADR
1648 002044          .WORD   IDMAINT+P0BR
1649 002046          .WORD   IDMAINT+P1BR
1650 002050          .WORD   IDMAINT+SBR
1651 002052          .WORD   IDMAINT+KSP
1652 002054          .WORD   IDMAINT+ESP
1653 002056          .WORD   IDMAINT+SSP
1654 002060          .WORD   IDMAINT+USP
1655 002062          .WORD   IDMAINT+ISP
1656 002064          .WORD   IDMAINT+FPDA
1657 002066          .WORD   IDMAINT+D.SV
1658 002070          .WORD   IDMAINT+Q.SV
1659 002072          .WORD   IDMAINT+TEMP0
1660 002074          .WORD   IDMAINT+TEMP1
1661 002076          .WORD   IDMAINT+TEMP2
1662 002100          .WORD   IDMAINT+TEMP3
1663 002102          .WORD   IDMAINT+TEMP4
1664 002104          .WORD   IDMAINT+TEMP5
1665 002106          .WORD   IDMAINT+TEMP6
1666 002110          .WORD   IDMAINT+TEMP7
1667 002112          .WORD   IDMAINT+TEMP8
1668 002114          .WORD   IDMAINT+TEMP9
1669 002116          .WORD   IDMAINT+PCBB
1670 002120          .WORD   IDMAINT+SCBB
1671 002122          .WORD   IDMAINT+P0LR
1672 002124          .WORD   IDMAINT+P1LR
1673 002126          .WORD   IDMAINT+SLR
1674 002130          ENDDAT
                ;-----------------------------------------------------------------------------

1675
```

```
1701            .SBTTL  T10     CIB INITIALIZE FUNCTION
                ;;*****************************************************************************
                ;++
                ;TEST   10      CIB INITIALIZE FUNCTION
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THAT AN LSI-11 RESET CLEARS ALL THE APPROPRIATE BITS
                ;       IN THE CONSOLE REGISTERS.
                ;
                ;       SUBTST 1 - CHECK THE ID CONTROL AND STATUS REG
                ;       SUBTST 2 - CHECK THE TO ID REG
                ;       SUBTST 3 - CHECK THE V BUS CONTROL REG
                ;       SUBTST 4 - CHECK THE MCR REG
                ;       SUBTST 5 - CHECK THE MCS REGISTER
                ;       SUBTST 6 - CHECK THE RECEIVER CS REG
                ;       SUBTST 7 - CHECK THE TRANSMITTER CS REG
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE INIT SIGNAL ON THE CIB MODULE.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED CONTENTS OF THE CIB REGISTER UNDER TEST
                ;             RECEIVED CONTENTS OF THE CIB REGISTER UNDER TEST
                ;
                ; SYNC POINT DESCRIPTION
                ;       THE SYNC POINT FOR EACH SUBTEST IS ON THE "RESETC" INSTRUCTION.
                ;--
                ;;*****************************************************************************
    002130      T10:
1705 002134             INITIALIZE
1706
1707 002136             SUBTEST
                ;//////////////////////////////////////////////////////////////////////////////
    002136      T10S1:
1708            ;+
1709            ; FIRST CHECK THAT THE IDCS REGISTER BITS 15, 7 AND <6:0> CLEAR
1710            ;-
1711
1712 002140             ERLOOP
1713 002142             LOADCA  IDCS,T30L1      ; LOAD THE IDCS WITH ALL ONE'S
1714 002152             RESETC                  ; EXECUTE THE RESET
1715 002154             CMPCAM  EQ,IDCS,T30M1,,T30L2 ; CHECK THAT THE BITS CLEARED
1716 002170             IFERROR ,CIB30          ; IDCS DID NOT INIT
1717
1718 002176             SUBTEST
                ;//////////////////////////////////////////////////////////////////////////////
    002176      T10S2:
1719            ;+
1720            ; CHECK THAT THE TO ID REGISTER BITS <31:0> CLEAR
1721            ;-
1722
1723 002200             ERLOOP
1724 002202             LOADCA  TOIDLO,T30L1    ; LOAD THE LOW 16 BITS
1725 002212             LOADCA  TOIDHI,T30L1    ; AND THE HIGH 16 BITS
1726 002222             RESETC                  ; EXECUTE THE RESET
1727 002224             CMPCAD  EQ,TOIDLO,T30L2 ; CHECK THAT THE REGISTER CLEARED
1728 002250             IFERROR ,CIB30          ; TO ID REGISTER DID NOT CLEAR
```

```
1729
1730 002256            SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
     002256     T10S3:
1731                    ;+
1732                    ; CHECK THAT BITS 2 AND 1 CLEAR IN THE V BUS CONTROL REG
1733                    ;-
1734
1735 002260            ERLOOP
1736 002262            LOADCA  VBCTRL,T30M2    ; SET THE BITS
1737 002272            RESETC                  ; EXECUTE THE RESET
1738 002274            CMPCAM  EQ,VBCTRL,T30M2,,T30L2 ; CHECK THAT THE BITS CLEARED
1739 002310            IFERROR ,CIB30          ; BITS 2 AND 1 DID NOT CLEAR
1740
1741 002316            SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
     002316     T10S4:
1742                    ;+
1743                    ; CHECK THAT THE APPROPRIATE BITS CLEAR IN THE MCR REGISTER
1744                    ;-
1745
1746 002320            ERLOOP
1747 002322            LOADCA  CONMCR,T30M3    ; SET THE BITS THAT SHOULD CLEAR
1748 002332            RESETC                  ; EXECUTE THE RESET
1749 002334            CMPCAM  EQ,CONMCR,T30M3,,T30L2 ; CHECK THAT THEY CLEARED
1750 002350            IFERROR ,CIB30          ; MCR DID NOT INITIALIZE
1751
1752                    ;+
1753                    ; NOW CHECK THAT "D MAINT RET" CLEARS
1754                    ;-
1755
1756 002356            ERLOOP
1757 002360            LOADCA  CONMCR,T30M3    ; LOAD THE BITS THAT SHOULD CLEAR
1758 002370            CLOCK   4               ; SET THE D MAINT RET BIT
1759 002374            RESETC                  ; EXECUTE THE RESET
1760 002376            TSTVB   T30L4           ; CHECK THAT D MAINT RET CLEARED
1761 002404            IFERROR ,CIB30          ; D MAINT RET DID NOT CLEAR
1762
1763 002412            SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
     002412     T10S5:
1764                    ;+
1765                    ; CHECK THAT BITS 5,6 & 11 CLEAR AND 12 SETS IN THE MCS REGISTER
1766                    ;-
1767
1768 002414            ERLOOP
1769 002416            SETPSW  340             ; ENSURE PRIORITY LEVEL 7 IS SET
1770 002422            LOADCA  CONMCS,T30M4    ; LOAD THE BITS THAT SHOULD CLEAR
1771 002432            RESETC                  ; EXECUTE THE RESET
1772 002434            CMPCAM  EQ,CONMCS,T30M5,,T30L3 ; CHECK THAT BIT 12 SETS AND 11,6, & 5 CLEARED
1773 002450            SETPSW  0               ; ALLOW INTERRUPTS AGAIN
1774 002454            IFERROR ,CIB30          ; MCS REGISTER INCORRECT
1775
1776 002462            SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
     002462     T10S6:
1777                    ;+
```

```
                        ; CHECK THAT THE RECEIVER DONE BIT CLEARS
                        ;-

002464                          ERLOOP
002466                          LOADCA  RXDNE,T30L5     ; SET THE DONE BIT
002476                          RESETC                  ; EXECUTE THE RESET
002500                          CMPCA   EQ,RXDNE,T30L2  ; CHECK THAT THE DONE BIT CLEARED
002512                          IFERROR ,CIB30
                        ;
002520                          SUBTEST
                        ;!!!
```

```
1840                    .SBTTL  SECTION NUMBER 04
     000000                     .PSECT  OVR04
                        .SBTTL  T11     CONSOLE REGISTER DUAL ADDRESSING
                        ;;*****************************************************************
                        ;++
                        ;TEST   11      CONSOLE REGISTER DUAL ADDRESSING
                        ;
                        ; TEST DESCRIPTION
                        ;       THIS TEST PERFORMS A DUAL ADDRESSING CHECK OF THE CONSOLE REGISTERS.
                        ;       THIS IS DONE BY WRITING A PATTERN INTO THE REGISTERS AND THEN CHECKING
                        ;       THAT THE REGISTERS STILL HAVE THE SAME PATTERN.
                        ;
                        ; LOGIC DESCRIPTION
                        ;       THIS TEST CHECKS THE Q BUS ADDRESS DECODE ON THE CIB MODULE.
                        ;
                        ; ERROR DESCRIPTION
                        ;       DATA: EXPECTED CONTENTS OF THE REGISTER
                        ;             RECIEVED CONTENTS OF THE REGISTER
                        ;
                        ; SYNC POINT DESCRIPTION
                        ;       THE EASIEST WAY TO DEBUG A FAILURE IS TO WRITE A ROUTINE TO REFERANCE
                        ;       THE REGISTER THAT DID NOT DECODE PROPERLY AND CHECK THE DECODE LOGIC
                        ;       TO SEE WHAT IS HAPPENING.
                        ;--
                        ;;*****************************************************************
     000034             T11:
1844 000040                     INITIALIZE
1845
1846                    ;+
1847                    ; FIRST PUT KNOWN DATA PATTERNS IN THE REGISTERS
1848                    ;-
1849
1850 000042                     LDIDREG CONTXD,T32L1    ; LOAD THE FROM ID REG
1851 000050                     LOADCA  RXDNE,T32L1+4   ; LOAD THE RXCS REG
1852 000060                     LOADCA  TXRDY,T32L1+6   ; AND THE TSCS
1853 000070                     LOADCA  TOIDLO,T32L1+10 ; THE TO ID LO
1854 000100                     LOADCA  TOIDHI,T32L1+12 ; AND THE HI BITS
1855 000110                     LOADCA  IDCS,T32L1+14   ; THE IDCS
1856 000120                     LOADCA  CONMCR,T32L1+16 ; THE MCR REG
1857 000130                     LOADCA  CONMCS,T32L1+20 ; AND THE MCS REG
1858
1859                    ;+
1860                    ; NOW CHECK THE REGISTERS
1861                    ;-
1862
1863 000140                     CMPCAD  ,FMIDLO,T32L1   ; CHECK THE FROM ID REG
1864 000164                     IFERROR ,CIB32
1865 000172                     CMPCA   ,RXDNE,T32L1+4  ; AND THE RXCS
1866 000204                     IFERROR ,CIB32
1867 000212                     CMPCA   ,TXRDY,T32L1+6  ; THE TXCS
1868 000224                     IFERROR ,CIB32
1869 000232                     CMPCAD  ,TOIDLO,T32L1+10 ; AND THE TO ID REG
1870 000256                     IFERROR ,CIB32
1871 000264                     CMPCA   ,IDCS,T32L3     ; CHECK THE IDCS
1872 000276                     IFERROR ,CIB32
1873 000304                     CMPCA   ,CONMCR,T32L1+16 ; CHECK THE MCR REG
1874 000316                     IFERROR ,CIB32
```

```
 1875 000324            CMPCAM  ,CONMCS,T32L1+20,,T32L1+20 ; AND THE MCS
 1876 000340            IFERROR ,CIB32
 1877 000346            CMPCAM  ,VBCTRL,T32M1,,T32L1+22
 1878 000362            IFERROR ,CIB32
 1879 000370            SKIP
 1880
 1881 000374  CIB32:    REPORT  <CIB>
 1882
 1883 000404            TESTDAT
                        ;-------------------------------------------------------------------------------
 1884 000410  T32L1:    .WORD   4,10,200,0,1,2,0,242,10000,200
 1885 000434  T32L2:    .WORD   IDMAINT+IDWRITE+CONTXD
 1886 000436  T32L3:    .WORD   37400
 1887 000440  T32M1:    .WORD   377
 1888 000442            ENDDAT
                        ;-------------------------------------------------------------------------------
 1889
```

```
1907                    .SBTTL  T12     WCS DATA REGISTER READ
                        ;;*****************************************************************************
                        ;++
                        ;TEST   12      WCS DATA REGISTER READ
                        ;
                        ; TEST DESCRIPTION
                        ;       THIS ROUTINE READS THE WCS DATA REGISTER AND TYPES THE
                        ;       NUMBER OF WCS MODULES PRESENT. IF THE PCS BITS (3-0) ARE NOT ZERO
                        ;       OR THERE ARE NO WCS MODULES PRESENT, EXECUTION RETURNS TO THE
                        ;       MONITOR. THE TEST SHOULD THEN BE LOOPED ON OR SINGLE INSTRUCTED
                        ;       TO DETERMINE WHY THE DATA REGISTER READ FAILED.
                        ;
                        ; LOGIC DESCRIPTION
                        ;       THIS TEST CHECKS THE ID BUS ADDRESS DECODE LOGIC AND PART OF THE
                        ;       ID BUS DATA INTERFACE ON THE WCS MODULE.
                        ;
                        ; SYNC POINT DESCRIPTION
                        ;       ADDRESS SYNC38--WCS DATA REGISTER IS READ
                        ;--
                        ;;*****************************************************************************
     000442             T12:
1911 000446                     INITIALIZE
1912 000450                     LOADCA  IDCS,T12L1      ; SETUP TO READ THE WCS DATA REGISTER
1913 000460                     CLOCK   3               ; STEP TO CPT3
1914 000464             SYNC38: CMPCAD  EQ,IDDATLO,T12L2 ; READ AND SAVE THE ID BUS DATA
1915 000510                     CLOCK   1               ; RETURN TO CPT0
1916 000514                     TYPSIZE                 ; TYPE THE WCS SIZE
1917
1918 000516                     TESTDAT
                        ;-------------------------------------------------------------------------------
1919 000522             T12L1:  .WORD   IDMAINT+USCDAT
1920 000524             T12L2:  .WORD   0
1921 000526                     ENDDAT
                        ;-------------------------------------------------------------------------------
1922
1923
```

```
                        .SBTTL  T13     INITIALIZE THE CONTROL STORE
                        ;;******************************************************************************
                        ;;++
                        ;TEST   13      INITIALIZE THE CONTROL STORE
                        ;
                        ; TEST DESCRIPTION
                        ;       THIS TEST (REALLY AN INITIALIZATION) WRITES ALL ZERO'S TO ALL
                        ;       WCS CONTROL STORE TO ENSURE THAT RANDOM PARITY ERRORS
                        ;       DON'T OCCUR IN FUTURE TESTS.
                        ;--
                        ;;******************************************************************************
     000526             T13:
     000532                     INITIALIZE
     000534                     LOOP    I,4096,8192,SIZEDEPENDENT
     000546                     BLKMIC  MICINIT,,0,1,I  ; FILL THE WCS WITH ALL ZERO'S
     000562                     ENDLOOP I

     000566                     TESTDAT
                        ;-------------------------------------------------------------------------------
     000572             MICINIT:.WORD   0               ; MICRO WORD OF ALL ZERO'S
     000574                     .WORD   0               ; ...
     000576                     .WORD   0               ; ...
     000600                     .WORD   0               ; ...
     000602                     .WORD   0               ; ...
     000604                     .WORD   0               ; ...
     000606                     ENDDAT
                        ;-------------------------------------------------------------------------------

```

```
1978            .SBTTL  T14     WCS ADDRESS REGISTER DATA INTEGRITY
                ;;*****************************************************************************
                ;++
                ;TEST   14      WCS ADDRESS REGISTER DATA INTEGRITY
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST FLOATS A ONE (IN A FIELD OF ZERO'S) AND A 0 (IN A
                ;       FIELD OF ONE'S) THRU THE WCS ADDRESS REGISTER.
                ;
                ;       SUBTST 1 - FLOAT A ZERO THRU THE ADDRESS REGISTER.
                ;       SUBTST 2 - FLOAT A ONE THRU THE ADDRESS REGISTER.
                ;       SUBTST 3 - CHECK THAT INIT CLEARS THE WCS ADDRESS REGISTER
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE ID BUS TRANCEIVERS AND THE WCS ADDRESS REGISTER
                ;       ON THE USC MODULE.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED CONTENTS OF THE WCS ADDRESS REGISTER
                ;             RECEIVED CONTENTS OF THE WCS ADDRESS REGISTER
                ;             LOOP COUNT -- INDICATES WHICH BIT POSITION IS UNDER TEST,
                ;               I.E. 1=BIT 0, 2=BIT 1, 3=BIT 2, ETC.
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC32--DATA IS WRITTEN TO THE REGISTER
                ;                  SYNC33--DATA IS READ BACK FROM THE REGISTER
                ;       SUBTST 2 - SYNC34--DATA IS WRITTEN TO THE REGISTER
                ;                  SYNC35--DATA IS READ BACK FROM THE REGISTER
                ;       SUBTST 3 - SYNC9A--WCS ADDRESS REG SHOULD CLEAR
                ;--
                ;;*****************************************************************************
    000606      T14:
1982 000612             INITIALIZE
1983
1984 000614             SUBTEST
                ;/////////////////////////////////////////////////////////////////////////////
    000614      T14S1:
1985            ;+
1986            ; FLOAT A ZERO IN A FIELD OF ONE'S
1987            ;-
1988
1989 000616             LOOP    I,1,16          ; GOING TO CHECK 16 BITS
1990 000630             FLTZRO  TMP1,I          ; GENERATE THE DATA PATTERN IN TMP1
1991 000636             ERLOOP                  ;
1992 000640     SYNC32: LDIDREG USCADR,TMP1     ; PUT THE DATA IN THE ADDRESS REGISTER
1993 000646     SYNC33: READID  USCADR          ; READ THE DATA BACK
1994 000652             CMPCA   EQ,IDREGLO,TMP1 ; CHECK AGAINST THE DATA THAT WAS LOADED
1995 000664             IFERROR 1,RPTUSC        ; IF ERROR, GO TO RPTUSC
1996 000672             ENDLOOP I               ; CONTINUE
1997
1998 000676             SUBTEST
                ;/////////////////////////////////////////////////////////////////////////////
    000676      T14S2:
1999            ;+
2000            ; FLOAT A ONE IN A FIELD OF ONE'S
2001            ;-
2002
```

T14 WCS ADDRESS REGISTER DATA INTEGRITY

```
2003 000700            LOOP    I,1,16          ; SET THE LOOP COUNT
2004 000712            FLTONE  TMP1,I          ; GENERATE THE TEST DATA
2005 000720            ERLOOP                  ;
2006 000722  SYNC34:   LDIDREG USCADR,TMP1     ; LOAD THE DATA PATTERN
2007 000730  SYNC35:   READID  USCADR          ; READ IT BACK
2008 000734            CMPCA   EQ,IDREGLO,TMP1 ; CHECK AGAINST TEST DATA
2009 000746            IFERROR 2,RPTUSC        ; IF ERROR, GO TO RPTUSC
2010 000754            ENDLOOP I               ; CONTINUE
2011
2012 000760            SUBTEST
                ;!!!
```

```
                .SBTTL  T15     WCS ADDRESS REGISTER COUNT LOGIC
                ;;*****************************************************************************
                ;++
                ;TEST   15      WCS ADDRESS REGISTER COUNT LOGIC
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST ENSURES WCS ADDRESS REGISTER BITS <14:00> COUNT
                ;       PROPERLY AND THAT WHEN BITS <14:13> ARE BOTH SET, NO
                ;       COUNT OCCURS.
                ;       THIS IS DONE BY LOADING THE REGISTER WITH A PATTERN OF
                ;       1FFF, WRITING THE DATA AND CHECKING THAT THE ADDRESS REG
                ;       INCREMENTED TO 3FFF, WRITING THE DATA REGISTER AGAIN AND CHECKING
                ;       THAT THE ADDRESS REGISTER INCREMENTED TO 5FFF, WRITING THE DATA
                ;       REGISTER AGAIN AND CHECKING THAT THE ADDRESS REGISTER ROLLED OVER
                ;       TO 0000.
                ;
                ;       SUBTST 1 - CHECK THAT BITS <14:00> COUNT PROPERLY
                ;       SUBTST 2 - CHECK THAT WHEN BITS <14:13> ARE BOTH SET, THE ADDRESS
                ;                  REGISTER DOES NOT COUNT.
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE LOGIC THAT CONTROLS THE INCREMENT OF THE WCS
                ;       ADDRESS REGISTER ON THE USC MODULE.
                ;
                ; ERROR DESCRIPTION
                ;       DATA:   EXPECTED CONTENTS OF THE WCS ADDRESS REGISTER
                ;            RECEIVED CONTENTS OF THE WCS ADDRESS REGISTER
                ;            LOOP COUNT - SUBTST 1 ONLY -- INDICATES THE EXPECTED PATTERN
                ;              CURRENTLY IN THE ADDRESS REG, I.E. 1=3FFF, 2=5FFF, 3=0000
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC36--WCS ADDRESS REGISTER SHOULD INCREMENT
                ;       SUBTST 2 - SYNC37--WCS ADDRESS SHOULD NOT INCREMENT
                ;--
                ;;*****************************************************************************
    001072      T15:
2072 001076             INITIALIZE
2073
2074 001100             SUBTEST
                ;//////////////////////////////////////////////////////////////////////////////
    001100      T15S1:
2075            ;+
2076            ; FIRST CHECK THAT THE ADDRESS REGISTER INCREMENTS CORRECTLY.
2077            ;-
2078
2079 001102             ERLOOP
2080 001104             LDIDREG USCADR,TMP2     ; INITIALIZE THE ADDRESS REGISTER
2081 001112             LOOP    I,1,3           ; SET THE LOOP COUNT
2082 001124     SYNC36: LDIDREG USCDAT,TMP2     ; REFERANCE THE DATA REGISTER
2083 001132             READID  USCADR          ; READ THE ADDRESS REGISTER BACK
2084 001136             CMPCA   EQ,IDREGLO,TMP3,I ; CHECK THE NEW CONTENTS OF THE ADR REG
2085 001150             IFERROR 3,USC2          ;
2086 001156             ENDLOOP I               ; CONTINUE
2087
2088 001162             SUBTEST
                ;//////////////////////////////////////////////////////////////////////////////
    001162      T15S2:
```

```
2089            ;+
2090            ; CHECK THAT NO INCREMENT OCCURS WHEN BITS <14:13>=11
2091            ;-
2092
2093  001164            ERLOOP
2094  001166            LDIDREG USCADR,TMP4     ; INITIALIZE THE ADDRESS REGISTER
2095  001174    SYNC37: LDIDREG USCDAT,TMP4     ; REFERANCE THE DATA REGISTER
2096  001202            READID  USCADR          ; GET THE ADDRESS REGISTER BACK
2097  001206            CMPCA   EQ,IDREGLO,TMP4 ; CHECK THAT THE REGISTER DIDN'T CHANGE
2098  001220            IFERROR 4,USC2
2099  001226            SKIP
2100
2101  001232    USC2:   REPORT  <USC>
2102
2103  001242            TESTDAT
                  ;-------------------------------------------------------------------------
2104  001246    TMP2:   .WORD   17777           ; USED TO CHECK INCREMENT
2105                    HEXDATA     1FFF
2106  001250    TMP3:   .WORD   37777,57777,0   ; VALUES OF THE ADR REG AFTER INCREMENTING
2107                    HEXDATA     3FFF,  5FFF, 0000
2108  001256    TMP4:   .WORD   77777           ; VALUE WHEN BITS<14:13>=11
2109                    HEXDATA     7FFF
2110  001260            ENDDAT
                  ;-------------------------------------------------------------------------
2111
```

```
2139            .SBTTL  T16     MICRO STACK DATA INTEGRITY
                ;;*****************************************************************************
                ;++
                ;TEST   16      MICRO STACK DATA INTEGRITY
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THE DATA INTEGRITY OF THE MICRO STACK BY FLOATING
                ;       A ONE AND A ZERO THROUGH LOCATION ZERO OF THE STACK. LOCATION
                ;       ZERO IS FORCED BY THE INIT FUNCTION.
                ;
                ;       SUBTST 1 - FLOAT A ZERO THRU THE MICRO STACK
                ;       SUBTST 2 - FLOAT A ONE THRU THE MICRO STACK
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE ID BUS INTERFACE TO THE MICRO STACK ON THE
                ;       USC MODULE.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED CONTENTS OF THE MICRO STACK
                ;             RECEIVED CONTENTS OF THE MICRO STACK
                ;             LOOP COUNT -- INDICATES WHAT BIT POSITION THE FLOATING ZERO
                ;               OR ONE IS CURRENTLY IN, I.E. 1=BIT 0, 2=BIT 1, ETC.
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC39--MICRO STACK IS WRITTEN
                ;                  SYNC3A--MICRO STACK IS READ
                ;       SUBTST 2 - SYNC3B--MICRO STACK IS WRITTEN
                ;                  SYNC3C--MICRO STACK IS READ
                ;--
                ;;*****************************************************************************
    001260      T16:
2143 001264             INITIALIZE
2144
2145 001266             SUBTEST
                ;/////////////////////////////////////////////////////////////////////////////
    001266      T16S1:
2146            ;+
2147            ; FLOAT A ZERO IN A FIELD OF ONE'S
2148            ;-
2149
2150 001270             LOOP    I,1,16          ; SET THE LOOP COUNT
2151 001302             FLTZRO  TMP5,I          ; GENERATE THE DATA PATTERN
2152 001310             ERLOOP
2153 001312             INITIALIZE              ; SET STACK POINTER AT ZERO
2154 001314     SYNC39: LDIDREG USCSTK,TMP5     ; WRITE THE STACK
2155 001322     SYNC3A: READID  USCSTK          ; READ THE STACK BACK
2156 001326             CMPCA   EQ,IDREGLO,TMP5 ; CHECK THE DATA WITH THAT WHICH WAS LOADED
2157 001340             IFERROR 5,USC3          ;
2158 001346             ENDLOOP I               ; CONTINUE
2159
2160 001352             SUBTEST
                ;/////////////////////////////////////////////////////////////////////////////
    001352      T16S2:
2161            ;+
2162            ; FLOAT A ONE IN A FIELD OF ZERO'S
2163            ;-
2164
```

```
2165 001354            LOOP    I,1,16          ; SET THE LOOP COUNT
2166 001366            FLTONE  TMP5,I          ; GENERATE THE TEST DATA
2167 001374            ERLOOP
2168 001376            INITIALIZE
2169 001400  SYNC3B:   LDIDREG USCSTK,TMP5     ; PUT THE DATA ON THE STACK
2170 001406  SYNC3C:   READID  USCSTK          ; GET THE DATA BACK
2171 001412            CMPCA   EQ,IDREGLO,TMP5 ; CHECK THE DATA THAT CAME BACK
2172 001424            IFERROR 6,USC3
2173 001432            ENDLOOP I               ; CONTINUE
2174 001436            SKIP
2175
2176 001442  USC3:     TSTVB   T16V1           ; IS STALL ACTIVE?
2177 001450            CHKPNT  1$              ; BRANCH IF NO
2178 001456            REPORT  <SBL>           ; "STALL" IS ACTIVE
2179 001466  1$:       REPORT  <USC,ICL>
2180
2181 001476            TESTDAT
                       ;-------------------------------------------------------------------------
2182 001502  TMP5:     .WORD   0,0
2183 001506  T16V1:    .WORD   1
2184 001510            VBUSG   0,15,0          ; STALL H
2185 001512            ENDDAT
                       ;-------------------------------------------------------------------------
2186
```

```
                        .SBTTL  T17     MICRO STACK DUAL ADDRESSING
                        ;;*****************************************************************************
                        ;++
                        ;TEST   17      MICRO STACK DUAL ADDRESSING
                        ;
                        ; TEST DESCRIPTION
                        ;       THIS TEST ENSURES THAT THE MICRO STACK POINTER INCREMENTS AND
                        ;       DECREMENTS CORRECTLY. THIS IS DONE BY FIRST TESTING THE
                        ;       INCREMENT AND DECREMENT SIGNALS ON USCJ AND THEN THE
                        ;       ADDERS, ON USCJ, ARE TESTED BY DOING 16 PUSHES, WITH DIFFERANT
                        ;       DATA, AND 16 POPS, CHECKING THE DATA ON EACH POP.
                        ;
                        ;       SUBTST 1 - CHECK THE INCREMENT AND DECREMENT FUNCTIONS BY WRITING
                        ;                  DIFFERANT DATA AT ADDRESS 0 AND 1, THEN CHECKING THAT THE
                        ;                  DATA IS READ BACK CORRECTLY.
                        ;       SUBTST 2 - CHECK THE STACK DUAL ADDRESSING BY WRITING THE ADDRESS OF
                        ;                  THE STACK LOCATION IN ITSELF, AND READING IT BACK AND
                        ;                  CHECKING IT. (THE DATA PATTERNS THAT ARE WRITTEN ARE
                        ;                  SUCH THAT EACH CHIP OF THE STACK RECEIVES THE ADDRESS
                        ;                  OF ITSELF)
                        ;
                        ; LOGIC DESCRIPTION
                        ;       THIS TEST CHECKS THE LOGIC THAT GENERATES THE MICRO STACK DECREMENT
                        ;       SIGNAL AND THE STACK POINTER ADDERS ON THE USC MODULE.
                        ;
                        ; ERROR DESCRIPTION
                        ;       DATA: EXPECTED CONTENTS OF THE MICRO STACK
                        ;             RECEIVED CONTENTS OF THE MICRO STACK
                        ;             LOOP COUNT - SUBTST 2 ONLY -- INDICATES THE STACK ADDRESS THAT
                        ;               HAS THE INCORRECT DATA, I.E. 1=ADR 0, 2=ADR 1, ETC.
                        ;
                        ;       NOTE: THE ERROR LOOP IN SUBTEST 2 INCLUDES THE WRITE OF ALL 16
                        ;             STACK ADDRESSES, AND THE READ OF ALL 16 ADDRESSES SO, IF
                        ;             ADDRESS 8 FAILED , FOR EXAMPLE, IT MIGHT BE NECESSARY TO
                        ;             SINGLE INSTRUCTION THROUGH THE LOOP.
                        ;
                        ; SYNC POINT DESCRIPTION
                        ;       SUBTST 1 - SYNC3D--LOCATION ZERO IS WRITTEN
                        ;                  SYNC3E--STACK SHOULD DECREMENT AND WRITE LOCATION 1
                        ;                  SYNC3F--READ LOCATION 1 AND STACK SHOULD INCREMENT
                        ;                  SYNC40--READ LOCATION ZERO AND STACK SHOULD INCREMENT
                        ;       SUBTST 2 - SYNC41--STACK IS DECREMENTED AND WRITTEN
                        ;                  SYNC42--STACK IS READ AND INCREMENTED
                        ;--
                        ;;*****************************************************************************
     001512             T17:
2233 001516                     SUBTEST
                        ;//////////////////////////////////////////////////////////////////////////////
     001516             T17S1:
2234                    ;+
2235                    ; CHECK THAT STACK DECREMENT AND INCREMENT WORKS CORRECTLY
2236                    ;-
2237
2238 001520                     ERLOOP
2239 001522                     INITIALIZE
2240 001524             SYNC3D: LDIDREG USCSTK,TMP6     ; WRITE SOME DATA AT ADDRESS 0
2241 001532             SYNC3E: LDIDREG USCSTK,TMP6+2   ; WRITE DIFFERENT DATA AT ADDRESS 1
```

```
2242 001540  SYNC3F: READID  USCSTK           ; READ ADDRESS 1 BACK
2243 001544           CMPCA   EQ,IDREGLO,TMP6+2 ; CHECK IT
2244 001556           IFERROR 7,USC4
2245 001564  SYNC40: READID  USCSTK           ; GET THE CONTENTS OF ADDRESS 0
2246 001570           CMPCA   EQ,IDREGLO,TMP6 ; CHECK THE DATA
2247 001602           IFERROR 10,USC4
2248
2249
2250 001610           SUBTEST
                    ;////////////////////////////////////////////////////////////////////////////////
     001610  T17S2:
2251                ;+
2252                ; NOW DO 16 PUSHES
2253                ;-
2254
2255 001612           ERLOOP
2256 001614           LOOP    I,1,16          ; SET THE LOOP COUNT
2257 001626  SYNC41: LDIDREG USCSTK,TMP7,I    ; WRITE THE ADDRESS IN EACH LOCATION
2258 001634           ENDLOOP I               ; CONTINUE
2259
2260                ;+
2261                ; NOW DO 16 POPS AND CHECK THE DATA
2262                ;-
2263
2264 001640           LOOP    I,16,1          ; SET THE LOOP COUNT
2265 001652  SYNC42: READID  USCSTK           ; GET THE TOP OF THE STACK
2266 001656           CMPCA   EQ,IDREGLO,TMP7,I ; CHECK AGAINST DATA THAT WAS WRITTEN
2267 001670           IFERROR 11,USC4
2268 001676           ENDLOOP I               ; CONTINUE
2269 001702           SKIP
2270
2271 001706  USC4:   REPORT  <USC>            ; CALLOUT MICRO SEQUENCER MODULE
2272
2273 001716           TESTDAT
                    ;-------------------------------------------------------------------------------
2274 001722  TMP6:    .WORD   125252,52525
2275                ;   HEXDATA      AAAA,  5555
2276 001726  TMP7:    .WORD   0,10421,21042,31463,42104,52525,63146,73567
2277                ;   HEXDATA      0, 1111, 2222, 3333, 4444, 5555, 6666, 7777
2278 001746           .WORD   104210,114631,125252,135673,146314,156735,167356,-1
2279                ;   HEXDATA      8888,  9999,  AAAA,  BBBB,  CCCC,  DDDD,  EEEE, FFFF
2280
2281 001766           ENDDAT
                    ;-------------------------------------------------------------------------------
2282
2325
```

```
2329            .SBTTL  T18     MAINTENANCE RETURN DATA INTEGRITY
                ;;*****************************************************************************
                ;++
                ;TEST   18      MAINTENANCE RETURN DATA INTEGRITY
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THE MAINTENANCE RETURN FUNCTION OF THE MICRO
                ;       SEQUENCER AND THE DATA INTEGRITY OF THE NUA BUS AND UPCSV
                ;       REGISTER. THIS IS DONE BY DOING MAINTENANCE RETURNS WITH A
                ;       FLOATING ONE AND ZERO PATTERNS ON THE MICRO STACK.
                ;
                ;       BEFORE THE FLOATING ONE AND ZERO PATTERNS ARE USED, A PATTERN
                ;       OF ALL ZERO'S AND ALL 1'S WILL BE TESTED. A FAILURE IN THESE
                ;       TWO TESTS WILL CAUSE V BUS SIGNALS TO BE EXAMINED TO DETERMINE
                ;       IF ANY BRANCH ENABLES ARE STUCK.
                ;       THE TEST OF ALL ZERO'S FIRST LOADS A MICRO WORD WITH A JUMP FIELD
                ;       OF 13FF(X), A SUB FIELD OF 3, AND A BEN FIELD OF 1F TO CHECK THAT
                ;       MAINTENANCE RETURN CLEARS THE LATCHES FOR THESE FIELDS
                ;       ON THE MICRO SEQUENCER.
                ;
                ;       SUBTST 1 - CHECK THE NUA BUS AND PC SAVE REG WITH ALL ONES
                ;       SUBTST 2 - CHECK THE NUA BUS AND PC SAVE REG WITH ALL ZERO'S.
                ;       SUBTST 3 - FLOAT A ZERO THRU THE UPC SAVE REGISTER
                ;       SUBTST 4 - FLOAT A ONE THRU THE UPCSV REGISTER
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE MICRO STACK INPUTS TO THE NUA BUS, THE NUA BUS
                ;       INPUT TO THE U MULTIPLEXOR, AND THE UPC BUS INPUT TO THE UPC SAVE
                ;       REGISTER ON THE USC MODULE.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED PC SAVE REG OR VBUS CHANNEL, BIT AND VALUE
                ;               OF THE NUA BUS.
                ;             RECEIVED PC SAVE REGISTER OR VBUS CHANNEL, BIT AND VALUE
                ;               OF THE NUA BUS.
                ;             LOOP COUNT - SUBTEST 3 AND 4 ONLY -- INDICATES WHAT BIT POSITION
                ;               THE FLOATING ONE OR ZERO IS CURRENTLY IN, I.E. 1=BIT 0,
                ;               2=BIT 1, 3=BIT2, ETC.
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC43--ALL ONES ARE ACTIVE ON THE NUA BUS
                ;                  SYNC44--NUA BUS IS CLOCKED INTO THE PC SAVE
                ;       SUBTST 2 - SYNC45--ALL ZERO'S ARE ACTIVE ON THE NUA BUS
                ;                  SYNC46--ALL ZERO'S ARE GATED INTO THE PC SAVE
                ;       SUBTST 3 - SYNC47--DATA PATTERN IS GATED INTO THE PCSAVE
                ;       SUBTST 4 - SYNC48--DATA PATTERN IS GATED INTO THE PCSAVE
                ;--
                ;;*****************************************************************************
    001766      T18:
2333 001772             INITIALIZE
2334
2335 001774             SUBTEST
                ;//////////////////////////////////////////////////////////////////////////////
    001774      T18S1:
2336            ;+
2337            ; FIRST EXECUTE A MAINTENANCE RETURN WITH ALL ONE'S.
2338            ;-
```

```
2339
2340 001776            LDIDREG USCSTK,TMP10+2  ; PUT THE ONE'S ON THE STACK
2341 002004            LOADCA  CONMCR,TMP11    ; SET THE MNT RTN BIT IN THE MCR
2342 002014            CLOCK   4               ; SET THE MNT RTN BIT IN THE SEQ
2343 002020            CLOCK   1               ; GO TO CPT50
2344 002024  SYNC43:   CLOCK   1               ; GO TO CTP125, NUA BUS IS ACTIVE
2345 002030            TSTVB   VB1A            ; CHECK THE NUA FOR ALL ONE'S
2346 002036            IFERROR ,USC5           ;
2347 002044  SYNC44:   CLOCK   2               ; LOAD THE PC SAVE REGISTER WITH THE NUA
2348 002050            CMPPCSV TMP10+2         ; CHECK THAT PC SAVE REG WAS LOADED
2349 002056            IFERROR ,USC5           ; BIT STUCK IN PC SAVE REG
2350
2351 002064            SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
     002064     T18S2:
2352               ;+
2353               ; CHECK THE ALL 0'S CONDITION & THAT MAINT RET CLEARS THE LATCHES
2354               ;-
2355
2356 002066            BLKMIC  T17L1,,11777,1  ; LOAD THE MICRO INSTRUCTION
2357 002102            ERLOOP
2358 002104            INITIALIZE              ; ENSURE CLOCK IS AT CPTO
2359 002106            FETCH   11777           ; FETCH THE MICRO INSTRUCTION
2360 002116            LOADCA  CONMCR,TMP11    ; SET THE MNT RETN BIT
2361 002126            LDIDREG USCSTK,TMP10    ; PUT THE ZERO'S ON THE STACK
2362 002134            CLOCK   1               ; GO TO CPT75
2363 002140  SYNC45:   CLOCK   1               ; GO TO CPT125
2364 002144            TSTVB   VB1B            ; CHECK NUA FOR ALL ZERO'S, IBUF EN DEASSERTED
2365                                           ; AND A BEN FIELD = 0.
2366 002152            IFERROR 13,USC5         ; IF FAILURE, GO TO USC5
2367 002160  SYNC46:   CLOCK   2               ; LOAD THE PC SAVE REGISTER
2368 002164            CMPPCSV TMP10           ; CHECK PC SAVE FOR ALL ZERO'S
2369 002172            IFERROR ,USC5           ; BIT STUCK IN PC SAVE REGISTER
2370
2371 002200            SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
     002200     T18S3:
2372               ;+
2373               ; NOW DO A FLOATING ZERO PATTERN
2374               ;-
2375
2376 002202            INITIALIZE              ; ENSURE CLOCK AT CPTO
2377 002204            LOOP    I,1,13          ; GOING TO CHECK 13 BITS OF MICRO ADDRESS
2378 002216            FLTZRO  TMP12,I         ; GENERATE THE TEST PATTERN
2379 002224            MASK    TMP12,TMP10+2   ; GET RID OF BITS 13,14, AND 15
2380 002232            ERLOOP
2381 002234            LDIDREG USCSTK,TMP12    ; PUT THE PATTERN ON THE MICRO STACK
2382 002242            LOADCA  CONMCR,T17L2    ; SET THE RETURN BIT
2383 002252            CLOCK   4               ; LATCH THE BIT IN THE MICRO SEQUENCER
2384 002256  SYNC47:   CLOCK   4               ; EXECUTE THE MAINTENANCE RETURN
2385 002262            CMPPCSV TMP12           ; CHECK AGAINST TEST DATA
2386 002270            IFERROR 14,USC5         ; IF ERROR GO TO USC5
2387 002276            ENDLOOP I               ; CONTINUE
2388
2389 002302            SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
     002302     T18S4:
```

```
2390          ;+
2391          ; NOW CHECK THE FLOATING ONE PATTERN
2392          ;-
2393
2394 002304            LOOP    I,1,13          ; SET THE LOOP COUNT
2395 002316            FLTONE  TMP12,I         ; GENERATE THE TEST PATTERN
2396 002324            MASK    TMP12,TMP10+2   ; MASK OFF BITS 13 THRU 15
2397 002332            ERLOOP
2398 002334            LDIDREG USCSTK,TMP12    ; PUT TEST PATTERN ON STACK
2399 002342            LOADCA  CONMCR,T17L2    ; SET MAINTENANCE RETURN BIT
2400 002352            CLOCK   4
2401 002356  SYNC48:   CLOCK   4               ; EXECUTE THE RETURN
2402 002362            CMPPCSV TMP12           ; CHECK THE PCSV REGISTER
2403 002370            IFERROR 15,USC5         ;
2404 002376            ENDLOOP I               ; CONTINUE
2405 002402            SKIP
2406
2407 002406  USC5:     TSTVB   T17V2                   ; CHECK THAT "UTRAP" IS NOT ACTIVE
2408 002414            CHKPNT  USC5E                   ; BRANCH IF OK
2409 002422            TSTVB   T17V21                  ; CHECK SBL MICRO TRAP SIGNALS
2410 002430            CHKPNT  USC5A                   ; GO TO USC5A IF OK
2411 002436            REPORT  <SBL>
2412
2413 002446  USC5A:    TSTVB   T17V22                  ; CHECK TBM MICRO TRAP SIGNALS
2414 002454            CHKPNT  USC5D                   ; GO TO USC5B IF OK
2415 002462            REPORT  <TBM>
2416
2417 002472  USC5D:    REPORT  <CEH,SBL>
2418
2419 002502  USC5E:    REPORT  <USC>
2420 002512            TESTDAT
             ;-------------------------------------------------------------------------------
2421 002516  VB1A:     .WORD   17
2422 002520            VBUSG   0,130,1         ; UPC 0
2423 002522            VBUSG   0,131,1         ; UPC 1
2424 002524            VBUSG   0,132,1         ; UPC 2
2425 002526            VBUSG   0,133,1         ; UPC 3
2426 002530            VBUSG   0,134,1         ; UPC 4
2427 002532            VBUSG   0,135,1         ; UPC 5
2428 002534            VBUSG   0,136,1         ; UPC 6
2429 002536            VBUSG   0,137,1         ; UPC 7
2430 002540            VBUSG   0,140,1         ; UPC 8
2431 002542            VBUSG   0,141,1         ; UPC 9
2432 002544            VBUSG   0,142,1         ; UPC 10
2433 002546            VBUSG   0,143,1         ; UPC 11
2434 002550            VBUSG   0,144,1         ; UPC 12
2435 002552            VBUSG   0,27,1          ; IBUF EN L
2436 002554            VBUSG   0,100,1         ; BEN <7:0> H
2437 002556  VB1B:     .WORD   17
2438 002560            VBUSG   0,130,0         ; UPC 0
2439 002562            VBUSG   0,131,0         ; UPC 1
2440 002564            VBUSG   0,132,0         ; UPC 2
2441 002566            VBUSG   0,133,0         ; UPC 3
2442 002570            VBUSG   0,134,0         ; UPC 4
2443 002572            VBUSG   0,135,0         ; UPC 5
2444 002574            VBUSG   0,136,0         ; UPC 6
2445 002576            VBUSG   0,137,0         ; UPC 7
```

```
2446 002600            VBUSG   0,140,0          ; UPC 8
2447 002602            VBUSG   0,141,0          ; UPC 9
2448 002604            VBUSG   0,142,0          ; UPC 10
2449 002606            VBUSG   0,143,0          ; UPC 11
2450 002610            VBUSG   0,144,0          ; UPC 12
2451 002612            VBUSG   0,27,1           ; IBUF EN L
2452 002614            VBUSG   0,100,1          ; BEN <7:0> H
2453
2454 002616  TMP10:    .WORD   0,17777          ; INITIAL TEST PATTERN
2455 002622  TMP11:    .WORD   SBC+MNTRTN       ; VALUE FOR MCR REGISTER
2456 002624  TMP12:    .WORD   0,0              ; WORKING LOCATION FOR FLOATING PATTERNS
2457 002630  T17V2:    .WORD   1
2458 002632            VBUSG   0,16,0           ; UTRAP L
2459 002634  T17V21:   .WORD   3
2460 002636            VBUSG   1,66,1           ; TIMEOUT TRAP L
2461 002640            VBUSG   1,67,1           ; RDS TRAP L
2462 002642            VBUSG   1,3,1            ; PAR ERR TRAP L
2463 002644  T17V22:   .WORD   3
2464 002646            VBUSG   1,70,1           ; TB PARITY UTRAP L
2465 002650            VBUSG   1,71,1           ; MISS UTRAP L
2466 002652            VBUSG   1,72,1           ; MBIT UTRAP L
2467 002654  T17V3:    .WORD   1
2468 002656            VBUSG   0,122,0          ; UTRAP (1) H
2469 002660  T17L1:
2470         ;*13FF:   SUB/SPEC,BEN/1F,J/13FF
2471 002660            .WORD   11777
2472 002662            .WORD   0
2473 002664            .WORD   4000
2474 002666            .WORD   600
2475 002670            .WORD   17477
2476 002672            .WORD   0
2477
2478 002674  T17L2:    .WORD   MNTRTN+CLRUWRD+SBC
2479 002676            ENDDAT
             ;-------------------------------------------------------------------------------
2480
```

```
                        .SBTTL  SECTION NUMBER 05
                                .PSECT  OVR05
                        .SBTTL  T19     MAINTENANCE RETURN MICRO STACK INCREMENT
                        ;;*****************************************************************
                        ;++
                        ;TEST   19      MAINTENANCE RETURN MICRO STACK INCREMENT
                        ;
                        ; TEST DESCRIPTION
                        ;       THIS TEST ENSURES THAT THE MICRO STACK POINTER INCREMENTS WHEN
                        ;       A MAINTENANCE RETURN FUNCTION IS EXECUTED.
                        ;       THIS IS DONE BY WRITING A PATTERN OF ONE'S AND ZERO'S ON THE STACK THEN,
                        ;       WRITING ALL ZERO'S ON THE STACK, EXECUTING A MNT RTN, AND CHECKING
                        ;       THE STACK FOR THE PATTERN OF ONE'S AND ZERO'S.
                        ;
                        ; LOGIC DESCRIPTION
                        ;       THIS TEST CHECKS PART OF THE LOGIC THAT GENERATES THE MICRO STACK
                        ;       INCREMENT SIGNAL ON THE USC MODULE.
                        ;
                        ; ERROR DESCRIPTION
                        ;       DATA: EXPECTED VALUE ON THE STACK
                        ;             RECEIVED VALUE ON THE STACK
                        ;
                        ; SYNC POINT DESCRIPTION
                        ;       ADDRESS SYNC49--STACK SHOULD INCREMENT
                        ;--
                        ;;*****************************************************************
     000034             T19:
2506 000040                     INITIALIZE
2507 000042                     LDIDREG USCSTK,TMP13    ; PUT CHECK WORD ON MICROSTACK
2508 000050                     LDIDREG USCSTK,TMP13+2  ; PUT ALL ZERO'S ON THE STACK
2509 000056                     LOADCA  CONMCR,TMP13+4  ; SET MNT RTN BIT
2510 000066             SYNC49: CLOCK   4               ; SET MNT RTN IN THE SEQ, ACTIVATES 'USTACK INC''
2511 000072                     CLOCK   4               ; POP THE STACK
2512 000076                     READID  USCSTK          ; GET THE TOP OF THE STACK
2513 000102                     CMPCA   EQ,IDREGLO,TMP13 ; SEE IF STACK INCREMENTED
2514 000114                     IFERROR 16,USC6         ;
2515 000122                     SKIP
2516 000126             USC6:   REPORT  <USC>           ; MNT RTN DID NOT INCREMENT MICRO STACK
2517
2518 000136                     TESTDAT
                        ;-----------------------------------------------------------------------
2519 000142             TMP13:  .WORD   52525
2520 000144                     .WORD   0
2521 000146                     .WORD   MNTRTN+CLRUWRD+SBC
2522 000150                     ENDDAT
                        ;-----------------------------------------------------------------------
2523
```

```
2547            .SBTTL  T1A     MICRO STACK WRITE DISABLE
                ;;*****************************************************************************
                ;++
                ;TEST   1A      MICRO STACK WRITE DISABLE
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST ENSURES THAT THE MICRO STACK WRITE ENABLE IS INHIBITED
                ;       DURING A MAINTENANCE RETURN.
                ;       THIS IS DONE BY WRITING A PATTERN OF ONE'S AND ZERO'S ON THE STACK,
                ;       THEN WRITING ALL ZERO'S ON THE STACK, SETTING THE MNT RTN LATCH
                ;       ON THE SEQUENCER, THEN TRYING TO WRITE THE MICRO STACK WITH ALL
                ;       ZERO'S. IF THE WRITE DISABLE FAILS, THE ZERO'S WILL BE WRITTEN
                ;       OVER THE PATTERN OF ONE'S AND ZERO'S.
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THAT THE MAINT RET SIGNAL IS CONNECTED TO THE
                ;       WRITE ENABLE LINES ON THE MICRO STACK.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED CONTENTS OF THE MICRO STACK
                ;             RECEIVED CONTENTS OF THE MICRO STACK
                ;
                ; SYNC POINT DESCRIPTION
                ;       ADDRESS SYNC4A--MICRO STACK WRITE ENABLE SHOULD BE DEASSERTED.
                ;--
                ;;*****************************************************************************
    000150      T1A:
2551 000154             INITIALIZE
2552 000156             LDIDREG USCSTK,TMP14    ; WRITE CHECK DATA ON STACK
2553 000164             LDIDREG USCSTK,TMP14+2  ; WRITE ADDRESS TO DO A MNT RTN ON
2554 000172             LOADCA  CONMCR,T16L1    ; SET MNT RTN BIT
2555 000202             CLOCK   4               ; SET MNT RTN LATCH ON USC
2556 000206             LOADCA  TOIDLO,TMP14+2  ; SETUP TO DO AN ID BUS WRITE TO
2557 000216             LOADCA  IDCS,TMP14+4    ; THE MICRO STACK
2558 000226 SYNC4A:     CLOCK   4               ; WRITE THE MICRO STACK, MNT RTN SHOULD
2559                                            ; INHIBIT THIS WRITE.
2560 000232             READID  USCSTK          ; GET THE STACK BACK
2561 000236             CMPCA   EQ,IDREGLO,TMP14 ; SEE IF CHECK DATA WAS OVERWRITTEN
2562 000250             IFERROR 17,USC7         ;
2563 000256             SKIP
2564
2565 000262 USC7:       REPORT  <USC>           ; MNT RTN DID NOT INHIBIT MICRO STACK WRITE
2566
2567 000272             TESTDAT
                ;-------------------------------------------------------------------------------
2568 000276 TMP14:      .WORD   125252,0
2569 000302             .WORD   IDMAINT+IDWRITE+USCSTK
2570 000304 T16L1:      .WORD   SBC+CLRUWRD+MNTRTN
2571 000306             ENDDAT
                ;-------------------------------------------------------------------------------
2572
2573
```

```
                        .SBTTL  T1B     WCS PARITY GENERATORS
                        ;;*****************************************************************************
                        ;++
                        ;TEST   1B      WCS PARITY GENERATORS
                        ;
                        ; TEST DESCRIPTION
                        ;       THIS TEST DOES A COMPLETE CHECK OF THE WCS PARITY GENERATORS.
                        ;       THIS IS DONE WITH 8 BIT PATTERNS. FIRST THE WCS ADDRESS REGISTER
                        ;       IS LOADED TO SET OR CLEAR THE PARITY INVERT BIT. THEN THE DATA PATTERN
                        ;       IS WRITTEN TO THE WCS DATA REGISTER. THE CLOCK IS STOPPED IN CPT100
                        ;       OF THIS WRITE AND THE PARITY GENERATE BIT IS EXAMINED ON THE V BUS
                        ;       FOR THE APPROPRIATE CONDITION.
                        ;
                        ;       ALL WCS MODULES PRESENT ARE TESTED.
                        ;
                        ; LOGIC DESCRIPTION
                        ;       THIS TEST CHECKS THE PARITY GENERATORS ON THE WCS MODULES.
                        ;
                        ; ERROR DESCRIPTION
                        ;       DATA: EXPECTED V BUS CHANNEL, BIT, AND VALUE (PARITY GENERATE BIT)
                        ;             RECEIVED V BUS CHANNEL, BIT, AND VALUE
                        ;             LOOP COUNT -- INDICATES WHICH DATA PATTERN (AND VALUE OF THE
                        ;               PARITY INVERT BIT) IS CURRENTLY BEING TESTED, I.E.
                        ;               1=PARITY INV SET & PATTERN A53C,ABEE
                        ;               2=PARITY INV CLR * PATTERN 4EEF,DCF4 , ETC.
                        ;             LOOP COUNTS FROM 9 THRU 16 ARE THE SAME AS COUNTS 1 THRU 8
                        ;             EXCEPT THE 6TH K OF CONTROL STORE IS BEING TESTED. LOOP COUNTS
                        ;             17 THRU 24 INDICATE THE 7TH K AND LOOP COUNTS 25 THRU 32
                        ;             8TH K.
                        ;
                        ; SYNC POINT DESCRIPTION
                        ;       ADDRESS SYNC4B--PARITY GENERATE SIGNAL IS ACTIVE
                        ;--
                        ;;*****************************************************************************
    000306              T1B:
2609 000312                     INITIALIZE
2610 000314                     LOOP    I,1,32,SIZE     ; GOING TO USE 8 PATTERNS PER WCS MODULE
2611 000326                     LDIDREG USCADR,TMP23,I  ; LOAD WCS ADDRESS AND PARITY INVERT BIT
2612 000334                     LOADCA  IDCS,TMPX1      ; SET UP TO WRITE THE DATA REGISTER
2613 000344                     LOADCA  TOIDLO,TMP24,I  ; LOAD LOWER 16 BITS OF PATTERN
2614 000354                     LOADCA  TOIDHI,TMPX24,I ; AND UPPER 16 BITS
2615 000364 SYNC4B:             CLOCK   2               ; STOP IN CPT100 OF THE WRITE
2616 000370                     ERLOOP
2617 000372                     READVB                  ; GET THE PARITY GENERATE BIT
2618 000376                     TSTVB   TMP25,I         ; CHECK THAT IT IS CORRECT
2619 000404                     IFERROR 22,1$           ;
2620 000412                     CLOCK   2               ; GO BACK TO CPTO
2621 000416                     ENDLOOP I               ; CONTINUE
2622 000422                     SKIP
2623 000426 1$:                 TSTVB   T1BV1           ; CHECK FOR "WCS WRITE CYCLE" ACTIVE
2624 000434                     CHKPNT  2$              ; GO TO 2$ IF OK
2625 000442                     REPORT  <USC>
2626 000452 2$:                 REPORT  <WCS,WCS2K,USC,PCS>
2627
2628 000462                     TESTDAT
                        ;-----------------------------------------------------------------------------
2629 000466 TMP23:
```

```
2630            ; 5TH K
2631 000466              .WORD   110000,10000,10000,10000,110000,10000,110000,110000 ; DATA FOR ADR REG
2632            ;   HEXDATA     9000,  1000,  1000, 1000, 9000,  1000, 9000,  9000
2633
2634            ; 6TH K
2635 000506              .WORD   112000,12000,12000,12000,112000,12000,112000,112000 ; DATA FOR ADR REG
2636            ;   HEXDATA     9400,  1400,  1400, 1400, 9400,  1400, 9400,  9400
2637
2638            ; 7TH K
2639 000526              .WORD   114000,14000,14000,14000,114000,14000,114000,114000 ; DATA FOR ADR REG
2640            ;   HEXDATA     9800,  1800,  1800, 1800, 9800,  1800, 9800,  9800
2641
2642            ; 8TH K
2643 000546              .WORD   116000,16000,16000,16000,116000,16000,116000,116000 ; DATA FOR ADR REG
2644            ;   HEXDATA     9C00,  1C00,  1C00, 1C00, 9C00,  1C00, 9C00,  9C00
2645
2646 000566  TMPX1:  .WORD   IDMAINT+IDWRITE+USCDAT
2647
2648            ;+
2649            ; THE FOLLOWING IS THE DATA FOR THE LOW 16 BITS OF THE DATA REGISTER
2650            ;-
2651
2652 000570  TMP24:  .WORD   125756          ; HEX ABEE
2653 000572           .WORD   156364          ; HEX DCF4
2654 000574           .WORD   164241          ; HEX E8A1
2655 000576           .WORD   41517           ; HEX 434F
2656 000600           .WORD   117673          ; HEX 9FBB
2657 000602           .WORD   73432           ; HEX 771A
2658 000604           .WORD   32125           ; HEX 3455
2659 000606           .WORD   0               ; HEX 0
2660            ; 6TH K
2661 000610           .WORD   125756          ; HEX ABEE
2662 000612           .WORD   156364          ; HEX DCF4
2663 000614           .WORD   164241          ; HEX E8A1
2664 000616           .WORD   41517           ; HEX 434F
2665 000620           .WORD   117673          ; HEX 9FBB
2666 000622           .WORD   73432           ; HEX 771A
2667 000624           .WORD   32125           ; HEX 3455
2668 000626           .WORD   0               ; HEX 0
2669            ; 7TH K
2670 000630           .WORD   125756          ; HEX ABEE
2671 000632           .WORD   156364          ; HEX DCF4
2672 000634           .WORD   164241          ; HEX E8A1
2673 000636           .WORD   41517           ; HEX 434F
2674 000640           .WORD   117673          ; HEX 9FBB
2675 000642           .WORD   73432           ; HEX 771A
2676 000644           .WORD   32125           ; HEX 3455
2677 000646           .WORD   0               ; HEX 0
2678            ; 8TH K
2679 000650           .WORD   125756          ; HEX ABEE
2680 000652           .WORD   156364          ; HEX DCF4
2681 000654           .WORD   164241          ; HEX E8A1
2682 000656           .WORD   41517           ; HEX 434F
2683 000660           .WORD   117673          ; HEX 9FBB
2684 000662           .WORD   73432           ; HEX 771A
2685 000664           .WORD   32125           ; HEX 3455
2686 000666           .WORD   0               ; HEX 0
```

```
                        ;+
                        ; THE FOLLOWING IS THE HIGH 16 BITS FOR THE DATA REGISTER
                        ;-

000670  TMPX24:  .WORD   122474          ; HEX A53C
000672           .WORD   47357           ; HEX 4EEF
000674           .WORD   136151          ; HEX BC69
000676           .WORD   14525           ; HEX 1955
000700           .WORD   53672           ; HEX 57BA
000702           .WORD   165723          ; HEX EBD3
000704           .WORD   171206          ; HEX B286
000706           .WORD   0               ; HEX 0
                ; 6TH K
000710           .WORD   122474          ; HEX A53C
000712           .WORD   47357           ; HEX 4EEF
000714           .WORD   136151          ; HEX BC69
000716           .WORD   14525           ; HEX 1955
000720           .WORD   53672           ; HEX 57BA
000722           .WORD   165723          ; HEX EBD3
000724           .WORD   171206          ; HEX B286
000726           .WORD   0               ; HEX 0
                ; 7TH K
000730           .WORD   122474          ; HEX A53C
000732           .WORD   47357           ; HEX 4EEF
000734           .WORD   136151          ; HEX BC69
000736           .WORD   14525           ; HEX 1955
000740           .WORD   53672           ; HEX 57BA
000742           .WORD   165723          ; HEX EBD3
000744           .WORD   171206          ; HEX B286
000746           .WORD   0               ; HEX 0
                ; 8TH K
000750           .WORD   122474          ; HEX A53C
000752           .WORD   47357           ; HEX 4EEF
000754           .WORD   136151          ; HEX BC69
000756           .WORD   14525           ; HEX 1955
000760           .WORD   53672           ; HEX 57BA
000762           .WORD   165723          ; HEX EBD3
000764           .WORD   171206          ; HEX B286
000766           .WORD   0               ; HEX 0

                        ;+
                        ; THE FOLLOWING IS THE EXPECTED VALUES OF THE PARITY GENERATE BIT
                        ;-

000770  TMP25:   .WORD   1
000772           VBUSG   2,50,0          ; PARITY GENERATE=WRONG
000774           .WORD   1
000776           VBUSG   2,50,1          ; PARITY GENERATE=CORRECT
001000           .WORD   1
001002           VBUSG   2,50,0          ; PARITY GENERATE=CORRECT
001004           .WORD   1
001006           VBUSG   2,50,1          ; PARITY GENERATE=CORRECT
001010           .WORD   1
001012           VBUSG   2,50,1          ; PARITY GENERATE=WRONG
001014           .WORD   1
001016           VBUSG   2,50,0          ; PARITY GENERATE=CORRECT
```

```
2744 001020            .WORD   1
2745 001022            VBUSG   2,50,0          ; PARITY GENERATE=WRONG
2746 001024            .WORD   1
2747 001026            VBUSG   2,50,1          ; PARITY GENERATE=WRONG
2748            ; 6TH K
2749 001030            .WORD   1
2750 001032            VBUSG   2,50,0          ; PARITY GENERATE=WRONG
2751 001034            .WORD   1
2752 001036            VBUSG   2,50,1          ; PARITY GENERATE=CORRECT
2753 001040            .WORD   1
2754 001042            VBUSG   2,50,0          ; PARITY GENERATE=CORRECT
2755 001044            .WORD   1
2756 001046            VBUSG   2,50,1          ; PARITY GENERATE=CORRECT
2757 001050            .WORD   1
2758 001052            VBUSG   2,50,1          ; PARITY GENERATE=WRONG
2759 001054            .WORD   1
2760 001056            VBUSG   2,50,0          ; PARITY GENERATE=CORRECT
2761 001060            .WORD   1
2762 001062            VBUSG   2,50,0          ; PARITY GENERATE=WRONG
2763 001064            .WORD   1
2764 001066            VBUSG   2,50,1          ; PARITY GENERATE=WRONG
2765            ; 7TH K
2766 001070            .WORD   1
2767 001072            VBUSG   2,50,0          ; PARITY GENERATE=WRONG
2768 001074            .WORD   1
2769 001076            VBUSG   2,50,1          ; PARITY GENERATE=CORRECT
2770 001100            .WORD   1
2771 001102            VBUSG   2,50,0          ; PARITY GENERATE=CORRECT
2772 001104            .WORD   1
2773 001106            VBUSG   2,50,1          ; PARITY GENERATE=CORRECT
2774 001110            .WORD   1
2775 001112            VBUSG   2,50,1          ; PARITY GENERATE=WRONG
2776 001114            .WORD   1
2777 001116            VBUSG   2,50,0          ; PARITY GENERATE=CORRECT
2778 001120            .WORD   1
2779 001122            VBUSG   2,50,0          ; PARITY GENERATE=WRONG
2780 001124            .WORD   1
2781 001126            VBUSG   2,50,1          ; PARITY GENERATE=WRONG
2782            ; 8TH K
2783 001130            .WORD   1
2784 001132            VBUSG   2,50,0          ; PARITY GENERATE=WRONG
2785 001134            .WORD   1
2786 001136            VBUSG   2,50,1          ; PARITY GENERATE=CORRECT
2787 001140            .WORD   1
2788 001142            VBUSG   2,50,0          ; PARITY GENERATE=CORRECT
2789 001144            .WORD   1
2790 001146            VBUSG   2,50,1          ; PARITY GENERATE=CORRECT
2791 001150            .WORD   1
2792 001152            VBUSG   2,50,1          ; PARITY GENERATE=WRONG
2793 001154            .WORD   1
2794 001156            VBUSG   2,50,0          ; PARITY GENERATE=CORRECT
2795 001160            .WORD   1
2796 001162            VBUSG   2,50,0          ; PARITY GENERATE=WRONG
2797 001164            .WORD   1
2798 001166            VBUSG   2,50,1          ; PARITY GENERATE=WRONG
2799
2800 001170  T1BV1:    .WORD   1
```

```
2801 001172          VBUSG   0,24,1          ; WCS WRITE CYCLE H
2802 001174          ENDDAT
                ;-------------------------------------------------------------------------------

2803
2804
```

```
                .SBTTL  T1C     CS BUS DATA INTEGRITY
                ;;*****************************************************************************
                ;++
                ;TEST   1C      CS BUS DATA INTEGRITY
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THE DATA INTEGRITY OF THE CS BUS BY FLOATING
                ;       A ONE AND A ZERO THRU A MICRO WORD, EXECUTING THE MICRO
                ;       WORD, AND CHECKING THE V BUS FOR PARITY ERRORS. ALL WCS MODULES
                ;       PRESENT ARE TESTED.
                ;
                ;       SUBTST 1 - FLOAT A ZERO THRU THE CS BUS
                ;       SUBTST 2 - FLOAT A ONE THRU THE CS BUS
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE ID BUS INTERFACE TO THE WCS MODULES, THE
                ;       DATA INTEGRITY OF THE WCS MEMORY CHIPS AND THE DATA INTEGRITY OF
                ;       THE CONTROL STORE (CS) BUS.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED V BUS CHANNEL, BIT AND VALUE
                ;             RECEIVED V BUS CHANNEL, BIT AND VALUE
                ;             LOOP COUNT - INDICATES WHICH BIT IN THE 32 BIT GROUP IS UNDER
                ;               TEST, I.E. 1=BIT 0, 2=BIT 1, 3=BIT 2, ETC.
                ;             LOOP COUNT - INDICATES WHICH 32 BIT GROUP IS UNDER
                ;               TEST, I.E. 1= BITS<31:0>, 2=BITS<63:32>, 3=BITS<95:64>
                ;               LOOP COUNTS 4 THRU 6 ARE THE SAME AS COUNTS 1 THRU 3
                ;               EXCEPT THE 6TH K OF CONTROL STORE IS BEING TESTED.
                ;               LOOP COUNTS 7 THRU 9 ARE FOR THE 7TH K AND LOOP COUNTS
                ;               10 THRU 12 THE 8TH K.
                ;
                ;       NOTE: THE EXPECTED AND RECEIVED V BUS CHANNEL INDICATES WHICH 32 BIT
                ;               GROUP HAS BAD PARITY IN IT, I.E. 102X=BITS<31:00>,
                ;               101X=BITS<63:32>, AND 100X=BITS<95:64>.
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC4C--TEST PATTERN IS ACTIVE ON THE CS BUS
                ;       SUBTST 2 - SYNC4D--TEST PATTERN IS ACTIVE ON THE CS BUS
                ;--
                ;;*****************************************************************************
    001174      T1C:
2846 001200             INITIALIZE
2847
2848 001202             SUBTEST
                ;////////////////////////////////////////////////////////////////////////////
    001202      T1CS1:
2849                    ;+
2850                    ; FIRST FLOAT A ZERO THRU THE CS BUS
2851                    ;-
2852
2853 001204             LOOP    J,1,12,SIZE     ; LOOP COUNT FOR THE 3 BANKS PER MODULE
2854 001216             LDIDREG USCADR,T1DL1,J  ; SELECT LOCATION ZERO
2855 001224             LOOP    K,1,3           ; INITIALIZE THE CONTENTS OF LOCATION 0
2856 001236             LDIDREG USCDAT,TMP102   ; ...
2857 001244             ENDLOOP K               ; ...
2858
2859 001250             LOOP    I,1,32          ; LOOP COUNT FOR THE BITS IN A BANK
```

```
2860 001262          FLTZRO  TMP101,I          ; GENERATE THE TEST PATTERN
2861 001270          ERLOOP
2862 001272          LDIDREG USCADR,TMP100,J ; LOAD THE BANK ADDRESS
2863 001300          LDIDREG USCDAT,TMP101   ; LOAD INTO THE SELECTED BANK
2864 001306          FETCH   T1DL1,J         ; EXECUTE THE MICRO WORD
2865 001316  SYNC4C: TSTVB   TMP103          ; CHECK THAT THERE WAS NO PARITY ERROR
2866 001324          IFERROR 30,1$           ;
2867 001332          ENDLOOP I               ; CONTINUE WITH THE NEXT BIT
2868 001336          ENDLOOP J               ; CONTINUE WITH THE NEXT BANK
2869 001342          SKIP    SUBTST
2870
2871 001346  1$:     LDIDREG USCADR,TMP100,J ; RELOAD THE ADDRESS REGISTER
2872 001354          LOADCA  TOIDLO,TMP101   ; SETUP TO WRITE THE DATA PATTERN
2873 001364          LOADCA  TOIDHI,TMP101+2 ; ...
2874 001374          LOADCA  IDCS,T1DL2      ; AND THE IDCS REGISTER
2875 001404          CLOCK   2               ; GO TO CPT125 OF THE WRITE
2876 001410          TSTVB   T1DV1,J         ; CHECK THE APPROPRIATE "CS WRITE" SIGNAL
2877 001416          CHKPNT  CSERR           ; BRANCH IF OK
2878 001424          REPORT  <USC>           ; "CS WRITE" SIGNAL NOT ACTIVE
2879
2880 001434  CSERR:  REPORT  <WCS,WCS2K,CSBUS>; CS BUS BIT(S) STUCK
2881
2882 001444          SUBTEST
             ;////////////////////////////////////////////////////////////////////////////
      001444 T1CS2:
2883          ;+
2884          ; NOW FLOAT A ONE THRU THE MICRO WORD
2885          ;-
2886
2887 001446          LOOP    J,1,12,SIZE     ; LOOP COUNT FOR 3 BANKS PER MODULE
2888 001460          LDIDREG USCADR,T1DL1,J  ; SELECT ADDRESS 0
2889 001466          LOOP    K,1,3           ; INITIALIZE THE MICRO WORD
2890 001500          LDIDREG USCDAT,TMPX25   ; ...
2891 001506          ENDLOOP K               ; ...
2892
2893 001512          LOOP    I,1,32          ; LOOP COUNT FOR 32 BITS
2894 001524          FLTONE  TMP101,I        ; GENERATE THE TEST PATTERN
2895 001532          ERLOOP
2896 001534          LDIDREG USCADR,TMP100,J ; LOAD THE BANK ADDRESS
2897 001542          LDIDREG USCDAT,TMP101   ; LOAD THE TEST PATTERN
2898 001550          FETCH   T1DL1,J         ; EXECUTE THE MICRO WORD
2899 001560  SYNC4D: TSTVB   TMP103          ; CHECK THAT THERE WAS NO PARITY ERROR
2900 001566          IFERROR 31,CSERR
2901 001574          ENDLOOP I               ; CONTINUE WITH NEXT BIT
2902 001600          ENDLOOP J               ; CONTINUE WITH NEXT BANK
2903
2904 001604          TESTDAT
             ;-------------------------------------------------------------------------
2905 001610  TMP100: .WORD   10000,30000,50000 ; BANK ADDRESS
2906          ;   HEXDATA     1000, 2000, 4000
2907
2908          ; 6TH K
2909 001616          .WORD   12000,32000,52000
2910          ;   HEXDATA     1400, 2400, 4400
2911
2912          ; 7TH K
2913 001624          .WORD   14000,34000,54000
```

```
2914            ;   HEXDATA      1800, 2800, 4800
2915
2916            ; 8TH K
2917 001632              .WORD   16000,36000,56000
2918            ;   HEXDATA      1C00, 2C00, 4C00
2919
2920 001640    TMP101: .WORD    0,0             ; WORKING LOCATION FOR TEST PATTERN
2921 001644    TMPX25: .WORD    0,0             ; USED TO INIT LOCATION ZERO
2922 001650    TMP102: .WORD    -1,-1           ; INITIALIZATION FOR ZERO FLOAT
2923 001654    T1DL1:  .WORD    10000
2924 001656            .WORD    10000
2925 001660            .WORD    10000
2926 001662            .WORD    12000
2927 001664            .WORD    12000
2928 001666            .WORD    12000
2929 001670            .WORD    14000
2930 001672            .WORD    14000
2931 001674            .WORD    14000
2932 001676            .WORD    16000
2933 001700            .WORD    16000
2934 001702            .WORD    16000
2935
2936 001704    TMP103: .WORD    3
2937 001706            VBUSG    1,2,0           ; CSPE0=0
2938 001710            VBUSG    1,1,0           ; CSPE1=0
2939 001712            VBUSG    1,0,0           ; CSPE2=0
2940
2941 001714    T1DL2:  .WORD    IDMAINT+IDWRITE+USCDAT
2942
2943 001716    T1DV1:  .WORD    1
2944 001720            VBUSG    0,21,1          ; CS WRITE (31:00) H
2945 001722            .WORD    1
2946 001724            VBUSG    0,22,1          ; CS WRITE (63:32) H
2947 001726            .WORD    1
2948 001730            VBUSG    0,23,1          ; CS WRITE (95:64) H
2949
2950 001732            .WORD    1
2951 001734            VBUSG    0,21,1          ; CS WRITE (31:00) H
2952 001736            .WORD    1
2953 001740            VBUSG    0,22,1          ; CS WRITE (63:32) H
2954 001742            .WORD    1
2955 001744            VBUSG    0,23,1          ; CS WRITE (95:64) H
2956
2957 001746            .WORD    1
2958 001750            VBUSG    0,21,1          ; CS WRITE (31:00) H
2959 001752            .WORD    1
2960 001754            VBUSG    0,22,1          ; CS WRITE (63:32) H
2961 001756            .WORD    1
2962 001760            VBUSG    0,23,1          ; CS WRITE (95:64) H
2963
2964 001762            .WORD    1
2965 001764            VBUSG    0,21,1          ; CS WRITE (31:00) H
2966 001766            .WORD    1
2967 001770            VBUSG    0,22,1          ; CS WRITE (63:32) H
2968 001772            .WORD    1
2969 001774            VBUSG    0,23,1          ; CS WRITE (95:64) H
2970
```

```
   2971 001776          ENDDAT
                ;-------------------------------------------------------------------------------

   2972
   2973
   2993
```

```
2997            .SBTTL  T1D     PCS PARITY CHECKERS
                ;;*****************************************************************************
                ;++
                ;TEST   1D      PCS PARITY CHECKERS
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST DOES A COMPLETE CHECK OF THE PCS PARITY CHECKERS.
                ;       THIS IS DONE WITH 8 DATA PATTERNS BY WRITING THE PATTERN INTO
                ;       WCS LOCATION 0, READING LOCATION 0 ONTO THE CS BUS, AND CHECKING THE
                ;       VALUE OF THE 3 CS PARITY ERROR BITS ON THE V BUS.
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE CS BUS PARITY CHECKERS ON THE PCS MODULE.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED V BUS CHANNEL, BIT, AND VALUE
                ;             RECEIVED V BUS CHANNEL, BIT, AND VALUE
                ;             LOOP COUNT -- INDICATES WHICH DATA PATTERN IS CURRENTLY UNDER
                ;               TEST, I.E. 1=0A86,9EA1, 2=7D3F,774F, ETC.
                ;
                ; SYNC POINT DESCRIPTION
                ;       ADDRESS SYNC4E--DATA PATTERN IS ON THE CS BUS AND PARITY CHECKERS
                ;                       ARE ACTIVE.
                ;--
                ;*****************************************************************************
    001776      T1D:
3001 002002             INITIALIZE
3002 002004             LOOP    I,1,8           ; GOING TO TEST 8 PATTERNS
3003 002016             ERLOOP
3004 002020             LDIDREG USCADR,TMP27,I  ; LOAD THE PARITY INVERT BIT
3005 002026             LDIDREG USCDAT,TMP26,I  ; LOAD ADR 0 BANK 0
3006 002034             LDIDREG USCDAT,TMP26,I  ; AND BANK 1
3007 002042             LDIDREG USCDAT,TMP26,I  ; AND BANK 2
3008 002050             FETCH   10000           ; EXECUTE ADDRESS 0
3009 002060     SYNC4E: TSTVB   TMP30,I         ; CHECK FOR THE CORRECT PARITY CONDITION
3010 002066             IFERROR 23,1$
3011 002074             ENDLOOP I               ; CONTINUE
3012 002100             SKIP
3013 002104     1$:     REPORT  <PCS>           ; EITHER CS BUS BAD OR PARITY CHECKERS BAD
3014
3015 002114             TESTDAT
                ;-----------------------------------------------------------------------------
3016
3017            ;+
3018            ; FOLLOWING IS THE WCS ADR REG DATA FOR EACH TIME THRU THE LOOP
3019            ;-
3020
3021 002120     TMP27:  .WORD   110000,10000,110000,10000,10000,110000,110000,10000
3022            ;   HEXDATA      9000, 8000,  9000,  8000, 8000, 9000,  9000,  8000
3023
3024            ;+
3025            ; THE FOLLOWING DATA IS THE TEST PATTERNS FOR THIS TEST:
3026            ;-
3027
3028            ;               LOWORD,HIWORD                   HIWRD,LOWRD
3029            ;               ------ ------                   ----- -----
3030
```

```
3031 002140  TMP26:  .WORD   117241,5206              ; HEX 0A86,9EA1
3032 002144          .WORD   73517,76477              ; HEX 7D3F,774F
3033 002150          .WORD   32673,157356             ; HEX DEEE,35BB
3034 002154          .WORD   125432,152150            ; HEX D468,AB1A
3035 002160          .WORD   156125,124527            ; HEX A957,DC55
3036 002164          .WORD   164756,73671             ; HEX 77B9,E9EE
3037 002170          .WORD   41364,121721             ; HEX A3D1,42F4
3038 002174          .WORD   0,0                      ; HEX 0000,0000
3039
3040              ;+
3041              ; FOLLOWING IS THE EXPECTED VALUE OF THE CS PARITY ERROR BITS:
3042              ;-
3043
3044 002200  TMP30:  .WORD   3                        ; LOOP COUNT = 1
3045 002202          VBUSG   1,2,1                    ; CSPE0=1
3046 002204          VBUSG   1,1,1                    ; CSPE1=1
3047 002206          VBUSG   1,0,1                    ; CSPE2=1
3048 002210          .WORD   3                        ; LOOP COUNT = 2
3049 002212          VBUSG   1,2,0                    ; CSPE0=0
3050 002214          VBUSG   1,1,0                    ; CSPE1=0
3051 002216          VBUSG   1,0,0                    ; CSPE2=0
3052 002220          .WORD   3                        ; LOOP COUNT = 3
3053 002222          VBUSG   1,2,1                    ; CSPE0=1
3054 002224          VBUSG   1,1,1                    ; CSPE1=1
3055 002226          VBUSG   1,0,1                    ; CSPE2=1
3056 002230          .WORD   3                        ; LOOP COUNT = 4
3057 002232          VBUSG   1,2,0                    ; CSPE0=0
3058 002234          VBUSG   1,1,0                    ; CSPE1=0
3059 002236          VBUSG   1,0,0                    ; CSPE2=0
3060 002240          .WORD   3                        ; LOOP COUNT = 5
3061 002242          VBUSG   1,2,0                    ; CSPE0=0
3062 002244          VBUSG   1,1,0                    ; CSPE1=0
3063 002246          VBUSG   1,0,0                    ; CSPE2=0
3064 002250          .WORD   3                        ; LOOP COUNT = 6
3065 002252          VBUSG   1,2,1                    ; CSPE0=1
3066 002254          VBUSG   1,1,1                    ; CSPE1=1
3067 002256          VBUSG   1,0,1                    ; CSPE2=1
3068 002260          .WORD   3                        ; LOOP COUNT = 7
3069 002262          VBUSG   1,2,1                    ; CSPE0=1
3070 002264          VBUSG   1,1,1                    ; CSPE1=1
3071 002266          VBUSG   1,0,1                    ; CSPE2=1
3072 002270          .WORD   3                        ; LOOP COUNT = 8
3073 002272          VBUSG   1,2,0                    ; CSPE0=0
3074 002274          VBUSG   1,1,0                    ; CSPE1=0
3075 002276          VBUSG   1,0,0                    ; CSPE2=0
3076 002300          ENDDAT
          ;-------------------------------------------------------------------------------
3077
```

```
3129            .SBTTL  SECTION NUMBER 06
     000000             .PSECT  OVR06
                .SBTTL  T1E     WCS DUAL ADDRESSING 5TH, 6TH, 7TH & 8TH K
                ;;*****************************************************************
                ;++
                ;TEST   1E      WCS DUAL ADDRESSING 5TH, 6TH, 7TH & 8TH K
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THE INDEPENDENCY OF THE UPC ADDRESS LINES ON THE
                ;       UPC BUS AND ON THE WCS MODULE. THIS IS DONE WITH THE FOLLOWING
                ;       SEQUENCE:
                ;               1) CLR 0                3) CLR 2
                ;                  WRITE 1                 WRITE 4
                ;                  CHECK 0                 CHECK 0
                ;                                          CHECK 1
                ;                                          CHECK 2
                ;
                ;               2) CLR 1                4) ETC.
                ;                  WRITE 2
                ;                  CHECK 0
                ;                  CHECK 1
                ;
                ;       A CLR CONSISTS OF WRITING GOOD PARITY, A WRITE CONSISTS OF WRITING
                ;       WRONG PARITY, AND A CHECK CONSISTS OF CHECKING FOR GOOD PARITY.
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE ADDRESS LINES ON THE UPC BUS, AND THE ADDRESS
                ;       LINE DRIVERS ON THE WCS MODULES.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED V BUS CHANNEL, BIT, AND VALUE
                ;             RECEIVED V BUS CHANNEL, BIT, AND VALUE
                ;             LOOP COUNT -- INDICATES WHICH ADDRESS WAS WRITTEN WITH BAD
                ;               PARITY, I.E. 1=ADR 1, 2=ADR 2, 3=ADR 4, 4=ADR 8, ETC
                ;             COUNTS 1 THRU 11 = 5TH K
                ;             COUNTS 12 THRU 22 = 6TH K
                ;             COUNTS 23 THRU 33 = 7TH K
                ;             COUNTS 34 THRU 44 = 8TH K
                ;
                ;             LOOP COUNT -- INDICATES WHICH ADDRESS WAS CHECKED FOR GOOD PARITY
                ;               AND DID NOT HAVE GOOD PARITY IN IT, I.E. 1=ADR 0, 2=ADR 1,
                ;               3=ADR 2, 4=ADR 4, 5=ADR 8, ETC.
                ;
                ;       NOTE: THE EXPECTED AND RECEIVED V BUS IDENTIFICATION SHOWS WHICH 32
                ;               BIT GROUP HAS THE INCORRECT PARITY, I.E. 102X=BITS<31:0>,
                ;               101X=BITS<63:32>, AND 100X=BITS<95:64>.
                ;
                ; SYNC POINT DESCRIPTION
                ;       ADDRESS SYNC4F--BAD PARITY IS WRITTEN
                ;       ADDRESS SYNC50--CS BUS IS CHECKED FOR GOOD PARITY
                ;
                ;       NOTE: SYNC50 IS INSIDE THE LOOP THAT CHECKS THE LOWER ORDER ADDRESSES
                ;               FOR GOOD PARITY SO, IT MAY BE NECESSARY TO STEP THRU THE
                ;               LOOP UNTIL THE FAILING ADDRESS IS BEING CHECKED.
                ;--
                ;;*****************************************************************
     000034     T1E:
```

```
                        INITIALIZE
                        LOOP    I,1,44,SIZE
000054                  LDIDREG USCADR,TMP31,I  ; LOAD ADDRESS TO WRITE GOOD PARITY
000062                  LOOP    K,1,3           ; WRITE 3 BANKS
000074                  LDIDREG USCDAT,T1BL1    ; WRITE GOOD PARITY
000102                  ENDLOOP K               ;

000106                  ERLOOP
000110                  LDIDREG USCADR,TMP32,I  ; LOAD ADDRESS TO WRITE WRONG PARITY
000116                  LOOP    K,1,3           ; LOAD 3 BANKS
000130  SYNC4F:         LDIDREG USCDAT,T1BL1    ; WRITE WRONG PARITY
000136                  ENDLOOP K               ;

000142                  LOOP    J,1,I           ; SETUP LOOP TO CHECK FOR GOOD PARITY
000154                  FETCH   TMP31,J         ; EXECUTE A MICRO WORD
000164  SYNC50:         TSTVB   T1BL2           ; CHECK FOR GOOD PARITY
000172                  IFERROR 24,1$           ;
000200                  ENDLOOP J               ; CONTINUE
000204                  ENDLOOP I               ; CONTINUE
000210                  SKIP

000214  1$:             REPORT  <WCS,WCS2K>     ; EITHER UPC BUS OR WCS ADDRESS LINES FAILED

000224                  TESTDAT
                ;-------------------------------------------------------------------------------
                ;+
                ; FOLLOWING ARE THE ADDRESSES USED TO WRITE GOOD PARITY:
                ;-

000230  TMP31:   .WORD   10000,10001,10002,10004,10010,10020,10040,10100,10200
                ;   HEXDATA     1000,  1001, 1002, 1004, 1008, 1010, 1020, 1040, 1080
000252              .WORD   10400,11000
                ;   HEXDATA      1100, 1200

                ; 6TH K
000256              .WORD   12000,12001,12002,12004,12010,12020,12040,12100,12200
                ;   HEXDATA      1400, 1401, 1402, 1404, 1408, 1410, 1420, 1440, 1480
000300              .WORD   12400,13000
                ;   HEXDATA      1500, 1600

                ; 7TH K
000304              .WORD   14000,14001,14002,14004,14010,14020,14040,14100,14200
                ;   HEXDATA      1800, 1801, 1802, 1804, 1808, 1810, 1820, 1840, 1880
000326              .WORD   14400,15000
                ;   HEXDATA      1900, 1A00

                ; 8TH K
000332              .WORD   16000,16001,16002,16004,16010,16020,16040,16100,16200
                ;   HEXDATA      1C00, 1C01, 1C02, 1C04, 1C08, 1C10, 1C20, 1C40, 1C80
000354              .WORD   16400,17000
                ;   HEXDATA      1D00, 1E00

                ;+
                ; FOLLOWING ARE THE ADDRESSES USED TO WRITE THE BAD PARITY:
                ;-
```

```
3189 000360  TMP32:  .WORD   110001,110002,110004,110010,110020,110040,110100
3190                 ;   HEXDATA      9001,  9002,  9004,  9008,  9010,  9020,  9040
3191 000376                  .WORD   110200,110400,111000,112000
3192                 ;   HEXDATA      9080,  9100,  9200,  9400
3193
3194                 ; 6TH K
3195 000406                  .WORD   112001,112002,112004,112010,112020,112040,112100,112200
3196                 ;   HEXDATA      9401,  9402,  9404,  9408,  9410,  9420,  9440,  9480
3197 000426                  .WORD   112400,113000,114000
3198                 ;   HEXDATA      9500,  9600, 9800
3199
3200                 ; 7TH K
3201 000434                  .WORD   114001,114002,114004,114010,114020,114040,114100,114200
3202                 ;   HEXDATA      9801,  9802,  9804,  9808,  9810,  9820,  9840,  9880
3203 000454                  .WORD   114400,115000,116000
3204                 ;   HEXDATA      9900,  9A00, 9C00
3205
3206                 ; 8TH K
3207 000462                  .WORD   116001,116002,116004,116010,116020,116040,116100,116200
3208                 ;   HEXDATA      9C01,  9C02,  9C04,  9C08,  9C10,  9C20,  9C40,  9C80
3209 000502                  .WORD   116400,117000,120000
3210                 ;   HEXDATA      9D00,  9E00, A000
3211
3212 000510  T1BL1:  .WORD   0,0
3213
3214                 ;+
3215                 ; FOLLOWING IS THE EXPECTED V BUS DATA.
3216                 ;-
3217
3218 000514  T1BL2:  .WORD   3
3219 000516          VBUSG   1,2,0           ; CSPE0
3220 000520          VBUSG   1,1,0           ; CSPE1
3221 000522          VBUSG   1,0,0           ; CSPE2
3222 000524          ENDDAT
                     ;-------------------------------------------------------------------------------
3223
```

```
   3225 000524                  FORCEOVR
                        .SBTTL  SECTION NUMBER 07
        000000                  .PSECT  OVR07
   3226
```

```
3264            .SBTTL  T1F     WCS DYNAMIC MEMORY TEST
                ;;*****************************************************************************
                ;++
                ;TEST   1F      WCS DYNAMIC MEMORY TEST
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST FIRST READS THE FIRST 4K OF PCS AND CHECKS FOR GOOD
                ;       PARITY.
                ;
                ;       THEN THIS TEST PERFORMS A PARTIAL MOVING INVERSIONS TEST ON THE 5TH
                ;       6TH, 7TH AND 8TH K OF WCS. THE TEST SEQUENCE IS AS FOLLOWS:
                ;
                ;       1) FILL WCS WITH ALL ZEROS
                ;       2) READ LOCATION 0 AND CHECK FOR GOOD PARITY
                ;       3) WRITE ALL 1'S IN LOCATION ZERO
                ;       4) READ LOCATION ZERO AND CHECK FOR GOOD PARITY
                ;       5) REPEAT 2,3, AND 4 FOR ALL WCS LOCATIONS
                ;       6) READ LOCATION 3FF AND CHECK FOR GOOD PARITY
                ;       7) WRITE LOCATION 3FF WITH ALL ZEROS
                ;       8) READ LOCATION 3FF AND CHECK FOR GOOD PARITY
                ;       9) REPEAT 6,7,8 FROM ADDRESSES 3FE TO 0
                ;
                ;       SUBTST 1 - CHECK THE PCS MODULE FOR PARITY ERRORS
                ;       SUBTST 2 - CHECK THE WCS MODULE
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE CS BUS DRIVERS ON THE PCS MODULE, AND THE
                ;       MEMORY CHIPS ON THE WCS MODULES.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED V BUS CHANNEL, BIT, AND VALUE
                ;             RECEIVED V BUS CHANNEL, BIT, AND VALUE
                ;             LOOP COUNT -- INDICATES WHICH PCS OR WCS ADDRESS IS
                ;               BEING TESTED, I.E. 1=ADR 0, 2=ADR 1, 3=ADR 2, 4=ADR 3, ETC.
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC51--PCS DATA IS ON THE CS BUS
                ;       SUBTST 2 - SYNC52--WCS DATA IS ON THE CS BUS
                ;--
                ;;*****************************************************************************
    000034      T1F:
3268 000040             INITIALIZE
3269
3270 000042             SUBTEST
                ;/////////////////////////////////////////////////////////////////////////////
    000042      T1FS1:
3271            ;+
3272            ; FIRST READ ALL OF PCS AND CHECK FOR GOOD PARITY
3273            ;-
3274
3275 000044              LOOP    I,1,4096        ; SET THE LOOP COUNT
3276 000056              ERLOOP
3277 000060              FETCH   0,I             ; READ THE LOCATION
3278 000070      SYNC51: TSTVB   MIVB            ; CHECK FOR GOOD PARITY
3279 000076              IFERROR ,PCSERR         ; PCS PARITY ERROR
3280 000104              ENDLOOP I               ; CONTINUE
3281 000110              SKIP    SUBTST          ; GO TO THE NEXT SUBTEST
```

```
                                ;////////////////////////////////////////////////////////////////////////////////
000124                          T1FS2:
                                ;+
                                ; NOW CHECK THE WCS
                                ;-

000126                          LOOP    I,1,4096,SIZE   ; FILL WCS WITH ZEROS
000140                          BLKMIC  MIMIC,,10000,1,I ; LOAD ZERO MICRO WORD
000154                          ENDLOOP I               ; CONTINUE

                                ;+
                                ; EXECUTE THE FOREWARD SEQUENCE (STEPS 2,3,4)
                                ;-

000160                          LOOP    I,1,4096,SIZE   ; SET THE LOOP COUNT
000172                          ERLOOP
000174                          FETCH   10000,I         ; READ A MICRO WORD
000204  SYNC52:                 TSTVB   MIVB            ; CHECK FOR GOOD PARITY
000212                          IFERROR ,WCS1           ; PARITY ERROR DETECTED
000220                          BLKMIC  MI1MIC,,10000,1,I ; WRITE ALL 1'S TO THE LOCATION
000234                          FETCH   10000,I         ; READ THE LOCATION
                                ;SYNC52
000244                          TSTVB   MIVB            ; CHECK FOR GOOD PARITY
000252                          IFERROR ,WCS1           ; PARITY ERROR DETECTED
000260                          ENDLOOP I               ; CONTINUE

                                ;+
                                ; NOW EXECUTE THE BACKWARD SEQUENCE (STEPS 6,7,8)
                                ;-

000264                          LOOP    I,4096,1,SIZE   ; SET THE LOOP COUNT
000276                          ERLOOP
000300                          FETCH   10000,I         ; READ THE LOCATION
                                ;SYNC52
000310                          TSTVB   MIVB            ; CHECK FOR GOOD PARITY
000316                          IFERROR ,WCS1           ; PARITY ERROR DETECTED
000324                          BLKMIC  MIMIC,,10000,1,I ; WRITE ZERO'S TO THE LOCATION
000340                          FETCH   10000,I         ; READ THE LOCATION
                                ;SYNC52
000350                          TSTVB   MIVB            ; CHECK FOR GOOD PARITY
000356                          IFERROR ,WCS1           ; PARITY ERROR DETECTED
000364                          ENDLOOP I               ; CONTINUE
000370                          SKIP

000374  WCS1:                  REPORT  <WCS,WCS2K>      ; WCS MEMORY FAILURE

000404                          TESTDAT
                                ;-------------------------------------------------------------------------------
000410  MIMIC:                  .WORD   0
000412                          .WORD   0
000414                          .WORD   0
000416                          .WORD   0
000420                          .WORD   0
```

```
3336 000422                .WORD   0
3337 000424  MI1MIC:       .WORD   -1
3338 000426                .WORD   -1
3339 000430                .WORD   -1
3340 000432                .WORD   -1
3341 000434                .WORD   -1
3342 000436                .WORD   -1
3343
3344                       ;+
3345                       ; FOLLOWING IS THE EXPECTED V BUS DATA.
3346                       ;-
3347
3348 000440  MIVB:         .WORD   3
3349 000442                VBUSG   1,2,0
3350 000444                VBUSG   1,1,0
3351 000446                VBUSG   1,0,0
3352 000450                ENDDAT
                           ;-------------------------------------------------------------------------------
3353
3354 000450                FORCEOVR
                   .SBTTL  SECTION NUMBER 08
     000000                .PSECT  OVR10
3355
3371
```

```
3375            .SBTTL  T20     NUA BUS DRIVERS
                ;;******************************************************************************
                ;++
                ;TEST   20      NUA BUS DRIVERS
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THAT THE "BUS NUA" DRIVERS ARE DISABLED WHEN
                ;       MAINTENANCE RETURN IS NOT ASCERTED. THIS IS DONE BY EXECUTING
                ;       A MICRO INSTRUCTION WITH A JMP FIELD OF ZERO AND ALL ONES ON THE
                ;       MICRO STACK. AFTER EXECUTION OF THE INSTRUCTION, THE UPC SAVE REG.
                ;       IS CHECKED FOR ALL ZERO'S.
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE "NAND" GATES THAT DRIVE THE NUA BUS DURING A
                ;       MAINTENANCE RETURN. IT CHECKS THAT THEY ARE NOT STUCK ON.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED UPC SAVE REG
                ;             RECEIVED UPC SAVE REG
                ;--
                ;;******************************************************************************
    000034      T20:
3379
3380 000040             INITIALIZE
3381
3382 000042             SUBTEST
                ;//////////////////////////////////////////////////////////////////////////////
    000042      T20S1:
3383            ;+
3384            ; CHECK THAT THE BUS DRIVERS ARE DISABLED WHEN MAINTENANCE RETURN IS
3385            ; DISABLED
3386            ;-
3387
3388 000044             BLKMIC  T40L1,,10000,1 ; LOAD THE MICRO INSTRUCTION
3389 000060             LDIDREG USCSTK,T40L2   ; PUT ALL ONES ON THE STACK
3390 000066             FETCH   10000          ; FETCH THE MICRO WORD
3391            ;SYNC
3392 000076             CLOCK   4              ; EXECUTE THE MICRO WORD
3393 000102             CMPPCSV T40L3          ; CHECK PC SAVE FOR ZERO
3394 000110             IFERROR ,USC40         ; BIT IS STUCK ON NUA
3395 000116             SKIP
3396
3397 000122     USC40:  REPORT  <USC>
3398
3399 000132             TESTDAT
                ;------------------------------------------------------------------------------
3400 000136     T40L1:
3401            ;*1000: NOP,J/0
3402 000136             .WORD   0
3403 000140             .WORD   0
3404 000142             .WORD   4000
3405 000144             .WORD   600
3406 000146             .WORD   74
3407 000150             .WORD   0
3408
3409 000152     T40L2:  .WORD   -1
3410 000154     T40L3:  .WORD   0
```

```
   3411 000156          ENDDAT
                ;-------------------------------------------------------------------------------
   3412
```

```
                        .SBTTL  T21     UBEN FIELD DECODE
                        ;;*****************************************************************************
                        ;++
                        ;TEST   21      UBEN FIELD DECODE
                        ;
                        ; TEST DESCRIPTION
                        ;       THIS TEST CHECKS THE LOGIC THAT DECODES THE BEN SIGNALS ON USCN.
                        ;       THIS IS DONE BY EXECUTING THE MICRO INSTRUCTION WITH THE BEN FIELD
                        ;       SET TO THE APPROPRIATE TEST PATTERN AND THEN EXAMINING THE BEN
                        ;       SIGNALS ON THE V BUS.
                        ;
                        ; LOGIC DESCRIPTION
                        ;       THIS TEST CHECKS THE UBEN FIELD LATCH AND THE LOGIC THAT GENERATES
                        ;       THE BEN ENABLES ON THE USC MODULE.
                        ;
                        ; ERROR DESCRIPTION
                        ;       DATA: EXPECTED V BUS CHANNEL, BIT, AND VALUE
                        ;             RECEIVED V BUS CHANNEL, BIT, AND VALUE
                        ;             LOOP COUNT -- INDICATES THE CURRENT BIT PATTERN IN THE UBEN
                        ;               FIELD, I.E. 1=00, 2=04, 3=08, 4=0C, 5=10, 6=14, ETC.
                        ;
                        ; SYNC POINT DESCRIPTION
                        ;       ADDRESS SYNC57--UBEN LATCH LATCHES AND BEN ENABLES ARE STABLE
                        ;--
                        ;*****************************************************************************
000156                  T21:
                        ;
000162                          LOOP    I,1,8           ; TAKES 8 PATTERNS TO TEST DECODE LOGIC
000174                          INITIALIZE              ; SET ROM NOP
000176                          LDIDREG USCADR,TMP35A   ; SELECT ADDRESS 1000(X)
000204                          LDIDREG USCDAT,T20L10   ; LOAD BANK 0
000212                          LDIDREG USCDAT,TMP35    ; LOAD BANK 1
000220                          LDIDREG USCDAT,TMP35,I  ; LOAD BANK 2
                        ;
000226                          ERLOOP
000230                          INITIALIZE
000232                          FETCH   10000,,NONOP
000242  SYNC57:                 TSTVB   TMP36,I         ; CHECK THE BEN ENABLES
000250                          IFERROR 27,1$           ;
000256                          ENDLOOP I               ; CONTINUE
000262                          SKIP
                        ;
000266  1$:                     REPORT  <USC>           ; BEN SIGNAL FAILED
                        ;
000276                          TESTDAT
                        ;-------------------------------------------------------------------------------
000302  T20L10:         .WORD   10000,0
000306  TMP35A:         .WORD   10000,0
000312  TMP35B:         .WORD   SBC
                        ;+
                        ; FOLLOWING ARE THE MICRO WORDS WITH THE DIFFERANT UBEN FIELDS.
                        ;-
                        ;
000314  TMP35:          .BYTE   0,0,0,0         ; BEN 00
000320                  .BYTE   0,4,0,0         ; BEN 04
000324                  .BYTE   0,10,0,0        ; BEN 08
000330                  .BYTE   0,14,0,0        ; BEN 0C
```

```
3469 000334            .BYTE   0,20,0,0          ; BEN 10
3470 000340            .BYTE   0,24,0,0          ; BEN 14
3471 000344            .BYTE   0,30,0,0          ; BEN 18
3472 000350            .BYTE   0,34,0,0          ; BEN 1C
3473
3474            ;+
3475            ; FOLLOWING IS THE EXPECTED V BUS DATA
3476            ;-
3477
3478 000354    TMP36:  .WORD   1                 ; LOOP COUNT = 1
3479 000356            VBUSG   0,77,0            ; BEN (13:10)
3480 000360            .WORD   3                 ; LOOP COUNT = 2
3481 000362            VBUSG   0,75,0            ; BEN (1B:14)
3482 000364            VBUSG   0,100,1           ; BEN (07:00)
3483 000366            VBUSG   0,101,0           ; BEN (0F:08)
3484 000370            .WORD   3                 ; LOOP COUNT = 3
3485 000372            VBUSG   0,75,0            ; BEN (1B:14)
3486 000374            VBUSG   0,100,0           ; BEN (07:00)
3487 000376            VBUSG   0,101,1           ; BEN (0F:08)
3488 000400            .WORD   1                 ; LOOP COUNT = 4
3489 000402            VBUSG   0,76,0            ; BEN (1F:1C)
3490 000404            .WORD   2                 ; LOOP COUNT = 5
3491 000406            VBUSG   0,75,0            ; BEN (1B:14)
3492 000410            VBUSG   0,77,1            ; BEN (13:10)
3493 000412            .WORD   4                 ; LOOP COUNT = 6
3494 000414            VBUSG   0,75,1            ; BEN (1B:14)
3495 000416            VBUSG   0,76,0            ; BEN (1F:1C)
3496 000420            VBUSG   0,77,0            ; BEN (13:10)
3497 000422            VBUSG   0,100,0           ; BEN (07:00)
3498 000424            .WORD   4                 ; LOOP COUNT = 7
3499 000426            VBUSG   0,75,1            ; BEN (1B:14)
3500 000430            VBUSG   0,76,0            ; BEN (1F:1C)
3501 000432            VBUSG   0,77,0            ; BEN (13:10)
3502 000434            VBUSG   0,101,0           ; BEN (0F:08)
3503 000436            .WORD   2                 ; LOOP COUNT = 8
3504 000440            VBUSG   0,75,0            ; BEN (1B:14)
3505 000442            VBUSG   0,76,1            ; BEN (1F:1C)
3506 000444            ENDDAT
           ;-------------------------------------------------------------------------------

3507
```

```
3538                .SBTTL  T22     USUB FIELD "CALL FUNCTION"
                    ;;*****************************************************************************
                    ;++
                    ;TEST   22      USUB FIELD "CALL FUNCTION"
                    ;
                    ; TEST DESCRIPTION
                    ;       THIS TEST CHECKS THE CALL FUNCTION OF THE USUB FIELD. FIRST,
                    ;       A TEST IS MADE TO ENSURE THE MICRO STACK DECREMENTS. THEN,
                    ;       THE DATA INTEGRITY OF THE MICRO STACK MUX IS CHECKED BY
                    ;       FLOATING A ONE AND ZERO THRU THE ADDRESS THAT GETS PUSHED BY
                    ;       THE "CALL".
                    ;
                    ;       SUBTST 1 - CHECK THAT THE MICRO STACK PUSHES
                    ;       SUBTST 2 - FLOAT A ZERO THRU THE ADDRESS THAT GETS PUSHED
                    ;       SUBTST 3 - FLOAT A ONE THRU THE ADDRESS THAT GETS PUSHED
                    ;
                    ; LOGIC DESCRIPTION
                    ;       THIS TEST CHECKS THE LOGIC THAT DECREMENTS THE MICRO STACK WHEN A
                    ;       CALL FUNCTION IS DECODED AND THE PC SAVE INPUTS TO THE MICRO STACK
                    ;       MULTIPLEXOR ON THE USC MODULE.
                    ;
                    ; ERROR DESCRIPTION
                    ;       DATA: EXPECTED CONTENTS OF THE MICRO STACK
                    ;             RECEIVED CONTENTS OF THE MICRO STACK
                    ;             LOOP COUNT - SUBTESTS 2 & 3 -- INDICATES THE BIT POSITION OF THE
                    ;               FLOATING ONE OR ZERO, I.E. 1=BIT 0, 2=BIT 1, 3=BIT 2, ETC.
                    ;
                    ; SYNC POINT DESCRIPTION
                    ;       SUBTST 1 - SYNC59--USTACK DEC IS ASSERTED
                    ;       SUBTST 2 - SYNC5A--PC SAVE IS WRITTEN ON THE MICRO STACK
                    ;       SUBTST 3 - SYNC5B--PC SAVE IS WRITTEN ON THE MICRO STACK
                    ;--
                    ;;*****************************************************************************
    000444          T22:
3542 000450             INITIALIZE
3543
3544 000452             SUBTEST
                    ;;;
```

```
                        SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
        000574  T22S2:
                ;+
                ; NOW FLOAT A ZERO THRU THE ADDRESS THAT GETS PUSHED
                ;-

                        LOOP    I,1,12          ; CHECK ALL 12 BITS
                        FLTZRO  TMP37,I         ; GENERATE THE TEST PATTERN
                        MASK    TMP37,MSK2      ; MAKE IT A LEGAL ADDRESS
                        INITIALIZE
                        LDIDREG USCADR,TMP37    ; SELECT THE TEST ADDRESS
                        LDIDREG USCDAT,TMP42    ; INITIALIZE THE TEST LOCATION
                        LDIDREG USCDAT,TMP42    ; ...
                        LDIDREG USCDAT,TMP40    ; PUT THE "CALL" IN BANK 2
                        ERLOOP
                        INITIALIZE
                        FETCH   TMP37,,NONOP    ; GET THE TEST INSTRUCTION
        SYNC5A:         CLOCK   4               ; EXECUTE THE SUBROUTINE CALL
                        READID  USCSTK          ; GET THE ADDRESS THAT WAS PUSHED
                        CMPCA   EQ,IDREGLO,TMP37 ; CHECK IF CORRECT ADDRESS WAS PUSHED
                        IFERROR 34,USC1         ; MICRO STACK MUX FAILED
                        ENDLOOP I               ; CONTINUE

                        SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
        000726  T22S3:
                ;+
                ; NOW FLOAT A ONE THRU THE ADDRESS THAT GETS PUSHED
                ;-

                        LOOP    I,1,11          ; CHECK ALL 11 BITS
                        INITIALIZE
                        BLKMIC  TMP42,,TMP37A,1,I ; INITIALIZE THE TEST LOCATION
                        ERLOOP
                        INITIALIZE
                        FETCH   TMP37A,I,NONOP  ; GET THE TEST LOCATION
        SYNC5B:         CLOCK   4               ; EXECUTE THE THE "CALL"
                        READID  USCSTK          ; GET THE ADDRESS THAT WAS PUSHED
                        CMPCA   EQ,IDREGLO,TMP37A,I ; CHECK THE ADDRESS THAT WAS PUSHED
                        IFERROR 35,USC1         ; MICRO STACK MUX FAILED
                        ENDLOOP I               ; CONTINUE
                        SKIP

        USC1:           REPORT  <USC>           ;

                        TESTDAT
                ;-------------------------------------------------------------------------------
                ;+
                ; FOLLOWING ARE THE ADDRESSES THAT ARE USED FOR THE FLOATING ONE PATTERN
                ;-

        TMP37A: .WORD   10001,10002,10004,10010,10020,10040,10100,10200,10400
                ;   HEXDATA      1001, 1002, 1004, 1008, 1010, 1020, 1040, 1080, 1100
                        .WORD   11000,10000
                ;   HEXDATA      1200, 1000
```

```
   3615 001076  TMP37:   .WORD   0,0              ; WORKING LOCATION FOR TEST PATTERN
   3616 001102  TMP40:   .WORD   1,0              ; BANK 2 DATA (SUB=CALL)
   3617 001106  TMP41:   .WORD   10000,0,0,0,0,0  ; JMP/0 MICRO INSTRUCTION
   3618 001122  TMP42:   .WORD   10000,0,0,0,1,0  ; SUB/CALL MICRO INSTRUCTION
   3619 001136  TMPX42:  .WORD   SBC
   3620 001140  TMP37B:  .WORD   10001
   3621 001142  MSK2:    .WORD   11777
   3622 001144            ENDDAT
                ;-------------------------------------------------------------------------------
   3623
```

T23   USUB FIELD "RETURN"

```
                        .SBTTL  T23     USUB FIELD "RETURN"
                        ;;*****************************************************************************
                        ;++
                        ;TEST   23      USUB FIELD "RETURN"
                        ;
                        ; TEST DESCRIPTION
                        ;       THIS TEST CHECKS THAT THE "RETURN" FUNCTION OF THE USUB FIELD
                        ;       CAUSES THE MICRO STACK TO BE POPPED AND THE ADDRESS ON THE STACK
                        ;       TO BE FETCHED.
                        ;       THIS IS DONE BY WRITING A ONE'S AND ZERO'S PATTERN ON THE STACK,
                        ;       WRITING THE TEST ADDRESS ON THE STACK, EXECUTING THE "RETURN"
                        ;       FUNCTION, AND CHECKING THE STACK FOR THE ONE'S AND ZERO'S PATTERN
                        ;       TO SEE THAT IT INCREMENT AND CHECKING THE PC SAVE REGISTER TO SEE
                        ;       THAT THE STACK WAS ENABLED ONTO THE NUA BUS.
                        ;
                        ; LOGIC DESCRIPTION
                        ;       THIS TEST CHECKS THE LOGIC THAT GENERATES THE USTACK INC SIGNAL WHEN
                        ;       A RETURN FUNCTION IS EXECUTED AND THE GATE THAT GENERATES THE
                        ;       RETURN SIGNAL THAT ENABLES THE MICRO STACK ONTO THE NUA BUS.
                        ;
                        ; ERROR DESCRIPTION
                        ;       DATA: EXPECTED CONTENTS OF THE MICRO STACK OR PC SAVE REGISTER
                        ;             RECEIVED CONTENTS OF THE MICRO STACK OR PC SAVE REGISTER
                        ;
                        ; SYNC POINT DESCRIPTION
                        ;       ADDRESS SYNC5C--USTACK INC SHOULD BE ASSERTED AND THE STACK SHOULD
                        ;                       BE ENABLED ONTO THE NUA BUS
                        ;--
                        ;;*****************************************************************************
    001144              T23:
3654 001150                     INITIALIZE
3655 001152                     LDIDREG USCADR,T20L1    ; SELECT ADDRESS 0
3656 001160                     LDIDREG USCDAT,T20L1    ; INITIALIZE LOCATION ZERO
3657 001166                     LDIDREG USCDAT,T20L1    ; ...
3658 001174                     LDIDREG USCDAT,TMP43    ; SET SUB FIELD TO "RETURN"
3659 001202                     ERLOOP
3660 001204                     INITIALIZE
3661 001206                     LDIDREG USCSTK,T20M1    ; PUT KNOWN DATA ON STACK
3662 001214                     LDIDREG USCSTK,TMP44    ; PUT ADDRESS TO RETURN TO ON STACK
3663 001222                     FETCH   10000,,NONOP    ; GET LOCATION 0
3664 001232             SYNC5C: CLOCK   4               ; EXECUTE THE RETURN
3665 001236                     LOADCA  CONMCR,T20L3    ; SET ROM NOP
3666 001246                     CMPPCSV TMP44           ; CHECK THAT STACK WAS ENABLED ONTO THE NUA BUS
3667 001254                     IFERROR 37,USC12        ; STACK WAS NOT GATED ONTO NUA BUS
3668 001262                     READID  USCSTK          ; GET THE TOP OF THE STACK
3669 001266                     CMPCA   EQ,IDREGLO,T20M1 ; CHECK TO SEE THAT STACK INCREMENTED
3670 001300                     IFERROR 36,USC12        ; MICRO STACK INCREMENT FAILED
3671
3672 001306                     SKIP
3673
3674 001312             USC12:  REPORT  <USC>           ;
3675
3676 001322                     TESTDAT
                        ;-----------------------------------------------------------------------------
3677 001326             TMP43:  .WORD   2,0             ; USUB FIELD GETS "RETURN"
3678 001332             TMP44:  .WORD   12525           ;
3679 001334             T20L1:  .WORD   10000,0,0,0,1,0
```

!23   USUB FIELD "RETURN"

```
   3680 001350  T20L2:  .WORD   SBC
   3681 001352  T20L3:  .WORD   CLRUWRD+SBC
   3682 001354  T20M1:  .WORD   11777
   3683 001356          ENDDAT
                ;-------------------------------------------------------------------------------
```

```
                .SBTTL  T24     USUB FIELD "SELECT SPECIFIER"
                ;;*****************************************************************************
                ;++
                ;TEST   24      USUB FIELD "SELECT SPECIFIER"
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THAT THE IBUF EN SIGNAL ASSERTS WHEN THE USUB FIELD
                ;       IS 3 (SELECT SPECIFIER). THIS IS DONE BY LOOKING AT THE BIT ON THE
                ;       V BUS.
                ;
                ;       SUBTST 1 - CHECK THAT IBUF EN DOES NOT ASSERT WHEN USUB = 1.
                ;       SUBTST 2 - CHECK THAT IBUF EN DOES NOT ASSERT WHEN USUB = 2.
                ;       SUBTST 3 - CHECK THAT IBUF EN ASSERTS WHEN USUB = 3
                ;       SUBTST 4 - CHECK THAT THE STACK DOES NOT CHANGE DURING A "SEL SPEC".
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE LOGIC ON THE USC MODULE THAT DECODES A USUB
                ;       FIELD = 3 AND CREATES THE SIGNAL "IBUF EN L".
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED V BUS CHANNEL, BIT, AND VALUE
                ;             RECEIVED V BUS CHANNEL, BIT AND VALUE
                ;
                ;       NOTE: FOR SUBTEST 4 THE DATA TYPEOUT IS THE CONTENTS OF THE STACK.
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNCAB--IBUF EN SHOULD BE DEASSERTED
                ;       SUBTST 2 - SYNCAC--IBUF EN SHOULD BE DEASSERTED
                ;       SUBTST 3 - SYNCAD--IBUF EN SHOULD BE ASSERTED
                ;       SUBTST 4 - SYNCAE--MICRO STACK SHOULD NOT DECREMENT
                ;--
                ;;*****************************************************************************
    001356      T24:
3717 001362             INITIALIZE
3718
3719 001364             SUBTEST
                ;/////////////////////////////////////////////////////////////////////////////
    001364      T24S1:
3720            ;+
3721            ; CHECK THAT IBUF EN DOES NOT ASSERT DURING A "CALL" FUNCTION
3722            ;-
3723
3724 001366             BLKMIC  T38L1,,10000,1  ; LOAD THE MICRO INSTRUCTION
3725 001402             ERLOOP
3726 001404             INITIALIZE
3727 001406             FETCH   10000,,NONOP    ; READUP THE INSTRUCTION
3728 001416             CLOCK   2               ; GO TO CPT100
3729 001422     SYNCAB: TSTVB   T38L4           ; CHECK THAT IBUF EN DID NOT ASSERT
3730 001430             CLOCK   2               ; SET CLOCK BACK TO CPT0
3731 001434             IFERROR ,USC38          ; IBUF EN ASSERTED
3732
3733 001442             SUBTEST
                ;/////////////////////////////////////////////////////////////////////////////
    001442      T24S2:
3734            ;+
3735            ; CHECK THAT IBUF EN DOES NOT ASSERT DURING A "RETURN" FUNCTION
3736            ;-
```

```
3737
3738 001444            INITIALIZE
3739 001446            BLKMIC  T38L5,,10000,1  ; LOAD THE MICRO INSTRUCTION
3740 001462            ERLOOP
3741 001464            INITIALIZE
3742 001466            FETCH   10000,,NONOP    ; READUP THE INSTRUCTION
3743 001476            CLOCK   2               ; GO TO CPT100
3744 001502  SYNCAC:   TSTVB   T38L4           ; CHECK THAT IBUF EN DID NOT ASSERT
3745 001510            CLOCK   2               ; SET CLOCK BACK TO CPT0
3746 001514            IFERROR ,USC38          ; IBUF EN ASSERTED
3747
3748 001522            SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
     001522     T24S3:
3749                   ;+
3750                   ; CHECK THAT IBUF EN ASSERTS WITH A "SELECT SPEC" FUNCTION.
3751                   ;-
3752
3753 001524            INITIALIZE
3754 001526            BLKMIC  T38L6,,10000,1  ; LOAD THE MICRO INSTRUCTION
3755 001542            ERLOOP
3756 001544            INITIALIZE
3757 001546            FETCH   10000,,NONOP    ; READUP THE INSTRUCTION
3758 001556            CLOCK   2               ; GO TO CPT100
3759 001562  SYNCAD:   TSTVB   T38L7           ; CHECK THAT IBUF EN DID ASSERT
3760 001570            IFERROR ,USC38          ; IBUF EN DID NOT ASSERT
3761
3762 001576            CLOCK   1               ; GO TO CPT150
3763 001602            TSTVB   T38L4           ; CHECK THAT IBUF EN DEASSERTED
3764 001610            CLOCK   1               ; GO TO CPT0
3765 001614            IFERROR ,USC38          ; IBUF EN DID NOT DEASSERT AT CPT150
3766
3767 001622            SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
     001622     T24S4:
3768                   ;+
3769                   ; CHECK THAT THE STACK DOES NOT INCREMENT OR DECREMENT DURING A "SELECT SPECIFIER"
3770                   ;-
3771
3772 001624            ERLOOP
3773 001626            INITIALIZE
3774 001630            LDIDREG USCSTK,T38L8    ; PUT KNOWN DATA ON THE STACK
3775 001636            FETCH   10000,,NONOP    ; READ THE INSTRUCTION
3776 001646  SYNCAE:   CLOCK   4               ; EXECUTE THE INSTRUCTION
3777 001652            READID  USCSTK          ; GET THE TOP OF THE STACK
3778 001656            CMPCA   EQ,IDREGLO,T38L8 ; CHECK THAT STACK DID NOT CHANGE
3779 001670            IFERROR ,USC38          ; STACK CHANGED DURING "SEL SPEC"
3780 001676            SKIP
3781
3782 001702  USC38:    REPORT  <USC>
3783
3784 001712            TESTDAT
                ;-------------------------------------------------------------------------------
3785 001716  T38L1:
3786                   ;*1000: SUB/CALL,J/1000
3787 001716            .WORD   10000
3788 001720            .WORD   0
```

```
3789 001722            .WORD   4000
3790 001724            .WORD   600
3791 001726            .WORD   75
3792 001730            .WORD   0
3793
3794 001732  T38L2:    .WORD   SBC
3795 001734  T38L3:    .WORD   10000
3796 001736  T38L4:    .WORD   1
3797 001740            VBUSG   0,27,1          ; IBUF EN L
3798 001742  T38L5:
3799           ;*1000: SUB/RET,J/1000
3800 001742            .WORD   10000
3801 001744            .WORD   0
3802 001746            .WORD   4000
3803 001750            .WORD   600
3804 001752            .WORD   76
3805 001754            .WORD   0
3806
3807 001756  T38L6:
3808           ;*1000: SUB/SPEC,J/1000
3809 001756            .WORD   10000
3810 001760            .WORD   0
3811 001762            .WORD   4000
3812 001764            .WORD   600
3813 001766            .WORD   77
3814 001770            .WORD   0
3815
3816 001772  T38L7:    .WORD   1
3817 001774            VBUSG   0,27,0          ; IBUF EN L
3818 001776  T38L8:    .WORD   125252
3819
3820 002000            ENDDAT
           ;-------------------------------------------------------------------------------
3821
```

```
                .SBTTL  T25     UJMP FIELD DATA INTEGRITY
                ;;*****************************************************************************
                ;++
                ;TEST   25      UJMP FIELD DATA INTEGRITY
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THE UJMP LATCH ON USCA BY FLOATING A ONE AND
                ;       A ZERO THRU THE UJMP FIELD OF THE MICRO WORD. THIS IS DONE BY
                ;       LOADING THE JMP MICRO WORDS INTO WCS, EXECUTING THEM UNDER SINGLE
                ;       BUS CYCLE, AND CHECKING THE UPCSV REGISTER.
                ;
                ;       SUBTST 1 - FLOAT A ZERO THRU THE UJMP FIELD
                ;       SUBTST 2 - FLOAT A ONE THRU THE UJMP FIELD
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE UJMP FIELD LATCH AND THE LATCHED INPUTS TO
                ;       THE NUA BUS ON THE USC MODULE.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED CONTENTS OF THE PC SAVE REGISTER
                ;             RECEIVED CONTENTS OF THE PC SAVE REGISTER
                ;             LOOP COUNT -- INDICATES THE BIT POSITION OF THE FLOATING ZERO
                ;               OR ONE, I.E. 1=BIT 0, 2=BIT 1, 3= BIT 2, 4=BIT 3, ETC.
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC53--UJMP FIELD IS LATCHED ON THE SEQUENCER
                ;                  SYNC54--LATCHED UJMP FIELD IS GATED ONTO THE NUA
                ;       SUBTST 2 - SYNC55--UJMP FIELD IS LATCHED ON THE SEQUENCER
                ;                  SYNC56--LATCHED UJMP FIELD IS GATED ONTO THE NUA
                ;--
                ;;*****************************************************************************
    002000      T25:
3854 002004             INITIALIZE
3855
3856 002006             SUBTEST
                ;//////////////////////////////////////////////////////////////////////////////
    002006      T25S1:
3857            ;+
3858            ; FLOAT A ZERO THRU THE JMP FIELD
3859            ;
3860            ; FIRST LOAD THE 10 JUMP INSTRUCTIONS INTO WCS
3861            ;-
3862
3863 002010             LOOP    I,1,10          ; GOING TO LOAD 10 JMP INSTRUCTIONS
3864 002022             LDIDREG USCADR,TMP34,I  ; SELECT THE ADDRESS
3865 002030             LDIDREG USCDAT,TMPX34,I ; WRITE THE JMP ADDRESS
3866 002036             LDIDREG USCDAT,XTMP34   ; PUT ZERO'S IN BANKS 1 AND 2
3867 002044             LDIDREG USCDAT,XTMP34   ; ...
3868 002052             ENDLOOP I               ; CONTINUE
3869
3870            ;+
3871            ; NOW EXECUTE THE INSTRUCTIONS AND CHECK THE PC SAVE REGISTER
3872            ;-
3873
3874 002056             INITIALIZE
3875 002060             LOOP    I,1,10          ; GOING TO EXECUTE 10 JMP INSTRUCTIONS
3876 002072             ERLOOP
```

T25 UJMP FIELD DATA INTEGRITY

```
3877 002074            INITIALIZE
3878 002076  SYNC53:  FETCH    TMP34,I,NONOP   ; FETCH THE TEST INSTRUCTION
3879 002106  SYNC54:  CLOCK    4               ; EXECUTE AN INSTRUCTION
3880 002112            CMPPCSV TMP34+2,I       ; CHECK RESULT OF JMP
3881 002120            IFERROR 26,USC11
3882 002126            ENDLOOP I
3883 002132            SKIP    SUBTST
3884
3885 002136  USC11:   REPORT  <USC>            ; JMP FIELD DATA PATH FAILED
3886
3887 002146            SUBTEST
             ;////////////////////////////////////////////////////////////////////////////////
      002146 T25S2:
3888         ;+
3889         ; FLOAT A ONE THRU THE JUMP FIELD
3890         ;
3891         ; FIRST LOAD 11 JUMP INSTRUCTIONS INTO WCS
3892         ;-
3893
3894 002150            INITIALIZE
3895 002152            LOOP    I,1,11
3896 002164            LDIDREG USCADR,TMP33,I  ; LOAD THE WCS ADDRESS TO WRITE IN
3897 002172            LDIDREG USCDAT,TMPX33,I ; LOAD THE JMP ADDRESS
3898 002200            LDIDREG USCDAT,TMP34-4  ; PUT ZERO'S IN BANKS 1 AND 2
3899 002206            LDIDREG USCDAT,TMP34-4  ; ...
3900 002214            ENDLOOP I               ; CONTINUE
3901
3902         ;+
3903         ; NOW START EXECUTING THE INSTRUCTIONS AND CHECKING THE PC SAVE REGISTER
3904         ;-
3905
3906 002220            LOOP    I,1,11          ; GOING TO EXECUTE 12 JMP'S
3907 002232            ERLOOP
3908 002234            INITIALIZE
3909 002236  SYNC55:  FETCH    TMP33,I,NONOP
3910 002246  SYNC56:  CLOCK    4               ; EXECUTE THE TEST INSTRUCTION
3911 002252            CMPPCSV TMP33+2,I       ; CHECK RESULT OF JMP
3912 002260            IFERROR 25,USC11
3913 002266            ENDLOOP I               ; CONTINUE
3914 002272            TESTDAT
             ;-------------------------------------------------------------------------------
3915         ;+
3916         ; FOLLOWING ARE THE WCS ADDRESS WHERE THE JUMP INSTRUCTIONS ARE LOADED
3917         ;-
3918
3919 002276  TMP33:    .WORD   10000,10001,10002,10004,10010,10020,10040,10100,10200
3920         ;    HEXDATA      1000, 1001, 1002, 1004, 1008, 1010, 1020, 1040, 1080
3921 002320            .WORD   10400,11000,0
3922         ;    HEXDATA      1100, 1200,0
3923
3924         ;+
3925         ; FOLLOWING ARE THE ADDRESSES THAT GET LOADED INTO THE ABOVE LOCATIONS
3926         ;-
3927
3928 002326  TMPX33:  .WORD   10001,0           ; HEX 1001
3929 002332            .WORD   10002,0         ; HEX 1002
3930 002336            .WORD   10004,0         ; HEX 1004
```

T25   UJMP FIELD DATA INTEGRITY

```
3931 002342            .WORD   10010,0         ; HEX 1008
3932 002346            .WORD   10020,0         ; HEX 1010
3933 002352            .WORD   10040,0         ; HEX 1020
3934 002356            .WORD   10100,0         ; HEX 1040
3935 002362            .WORD   10200,0         ; HEX 1080
3936 002366            .WORD   10400,0         ; HEX 1100
3937 002372            .WORD   11000,0         ; HEX 1200
3938 002376            .WORD   0,0
3939
3940            ;+
3941            ; FOLLOWING ARE THE ADDRESSES THAT GET LOADED WITH JMP INSTRUCTIONS
3942            ; FOR SUBTST 2
3943            ;-
3944
3945 002402  TMP34:  .WORD   10000,11776,11775,11773,11767,11757,11737,11677
3946            ;    HEXDATA     1000, 13FE, 13FD, 13FB, 13F7, 13EF, 13DF, 13BF
3947 002422            .WORD   11577,11377,10777
3948            ;    HEXDATA     137F, 12FF, 11FF
3949
3950            ;+
3951            ; FOLLOWING ARE THE JMP ADDRESS THAT GET LOADED AT THE ABOVE WCS ADDRESSES
3952            ;-
3953
3954 002430  TMPX34: .WORD   11776,0         ; HEX 13FE
3955 002434            .WORD   11775,0         ; HEX 13FD
3956 002440            .WORD   11773,0         ; HEX 13FB
3957 002444            .WORD   11767,0         ; HEX 13F7
3958 002450            .WORD   11757,0         ; HEX 13EF
3959 002454            .WORD   11737,0         ; HEX 13DF
3960 002460            .WORD   11677,0         ; HEX 13BF
3961 002464            .WORD   11577,0         ; HEX 137F
3962 002470            .WORD   11377,0         ; HEX 12FF
3963 002474            .WORD   10777,0         ; HEX 11FF
3964
3965 002500  XTMP34: .WORD    0,0
3966 002504  TMP34Z: .WORD    SBC
3967 002506          ENDDAT
            ;-------------------------------------------------------------------------------
3968
```

```
                        .SBTTL  SECTION NUMBER 09
                        .PSECT  OVR11
                        .SBTTL  T26     MICRO BREAK REGISTER DATA INTEGRITY
                        ;;*****************************************************************************
                        ;++
                        ;TEST   26      MICRO BREAK REGISTER DATA INTEGRITY
                        ;
                        ; TEST DESCRIPTION
                        ;       THIS TEST FLOATS A ONE AND A ZERO THRU THE MICRO BREAK REGISTER.
                        ;
                        ;       SUBTST 1 - FLOAT A ZERO THRU THE REGISTER
                        ;       SUBTST 2 - FLOAT A ONE THRU THE REGISTER
                        ;       SUBTST 3 - CHECK THAT SYSTEM INIT DOES NOT CLEAR THE UBREAK REG
                        ;
                        ; LOGIC DESCRIPTION
                        ;       THIS TEST CHECKS THE ID BUS INTERFACE TO THE MICRO BREAK REGISTER
                        ;       AND THE REGISTER ITSELF ON THE USC MODULE.
                        ;
                        ; ERROR DESCRIPTION
                        ;       DATA: EXPECTED CONTENTS OF THE MICRO BREAK REGISTER
                        ;             RECEIVED CONTENTS OF THE MICRO BREAK REGISTER
                        ;             LOOP COUNT -- INDICATES WHAT BIT POSITION THE FLOATING ONE
                        ;               OR ZERO IS IN, I.E. 1=BIT 0, 2=BIT 1, 3=BIT 2, ETC.
                        ;
                        ; SYNC POINT DESCRIPTION
                        ;       SUBTST 1 - SYNC5D--MICRO BREAK REGISTER IS CLOCKED
                        ;                  SYNC5E--MICRO BREAK IS ENABLED ONTO ID BUS
                        ;       SUBTST 2 - SYNC5F--MICRO BREAK REG IS CLOCKED
                        ;                  SYNC60--MICRO BREAK REG IS ENABLED ONTO ID BUS
                        ;--
                        ;;*****************************************************************************
000034                  T26:
3999 000040                     INITIALIZE
4000
4001 000042                     SUBTEST
                        ;//////////////////////////////////////////////////////////////////////////////
000042                  T26S1:
4002                    ;+
4003                    ; FIRST FLOAT A ZERO THRU THE REGISTER
4004                    ;-
4005
4006 000044                     LOOP    I,1,13          ; GOING TO CHECK 13 BITS
4007 000056                     FLTZRO  T21L1,I         ; GENERATE THE TEST PATTERN
4008 000064                     MASK    T21L1,T21M1     ; MASK IT TO 13 BITS
4009 000072                     ERLOOP
4010 000074                     LOADCA  TOIDLO,T21L1    ; LOAD THE LOW 16 BITS OF THE TEST PATTERN
4011 000104                     LOADCA  IDCS,T21L2      ; SETUP TO WRITE THE MICRO BREAK
4012 000114                     CLOCK   2               ; GO TO CPT100
4013 000120             SYNC5D: CLOCK   1               ; MICRO BREAK REG CLOCKS
4014 000124                     CLOCK   1               ; GO TO CPT0
4015 000130             SYNC5E: READID  USCBRK          ; READ THE REGISTER BACK
4016 000134                     CMPCA   EQ,IDREGLO,T21L1 ; CHECK THE RECEIVED DATA
4017 000146                     IFERROR 40,USC13        ; MICRO BREAK REGISTER FAILED
4018 000154                     ENDLOOP I               ; CONTINUE
4019 000160                     SKIP    SUBTST
4020
4021 000164             USC13:  REPORT  <USC>
```

```
4022
4023 000174            SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
     000174     T26S2:
4024               ;+
4025               ; NOW FLOAT A ONE THRU THE MICRO BREAK REGISTER
4026               ;-
4027
4028 000176            LOOP    I,1,13
4029 000210            FLTONE  T21L1,I         ; GENERATE THE TEST PATTERN
4030 000216            ERLOOP
4031 000220            LOADCA  TOIDLO,T21L1    ; LOAD TEST PATTERN
4032 000230            LOADCA  IDCS,T21L2      ; SETUP TO WRITE THE BREAK REGISTER
4033 000240            CLOCK   2               ; GO TO CPT100
4034 000244  SYNC5F:   CLOCK   1               ; CLOCK THE MICRO BREAK REG
4035 000250            CLOCK   1               ; GO TO CPTO
4036 000254  SYNC60:   READID  USCBRK          ; GET THE DATA BACK
4037 000260            CMPCA   EQ,IDREGLO,T21L1 ; CHECK THAT THE DATA IS THE SAME
4038 000272            IFERROR 41,USC13        ; MICRO BREAK REGISTER FAILED
4039 000300            ENDLOOP I               ; CONTINUE
4040
4041 000304            SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
     000304     T26S3:
4042               ;+
4043               ; NOW CHECK THAT SYSTEM INIT DOES NOT CLEAR THE UBREAK REGISTER
4044               ;-
4045
4046 000306            LDIDREG USCBRK,T21L3    ; LOAD THE BREAK WITH ALL ONES
4047 000314            LOADCA  CONMCR,T21L4    ; SET SYS INIT
4048 000324            LOADCA  CONMCR,T21L5    ; CLEAR SYSTEM INIT
4049 000334            READID  USCBRK          ; READ THE BREAK REGISTER
4050 000340            CMPCA   ,IDREGLO,T21L3  ; CHECK THAT BREAK DIDN'T CHANGE
4051 000352            IFERROR ,USC13
4052
4053 000360            TESTDAT
                ;-------------------------------------------------------------------------------
4054 000364  T21L1:    .WORD   0,0
4055 000370  T21L2:    .WORD   IDMAINT+IDWRITE+USCBRK
4056 000372  T21L3:    .WORD   17777
4057 000374  T21L4:    .WORD   INIT+CLRUWRD+SBC
4058 000376  T21L5:    .WORD   CLRUWRD+SBC
4059 000400  T21M1:    .WORD   17777
4060 000402            ENDDAT
                ;-------------------------------------------------------------------------------
4061
```

```
4111            .SBTTL  T27     MICRO BREAK COMPARITORS
                ;;*****************************************************************************
                ;++
                ;TEST   27      MICRO BREAK COMPARITORS
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THAT THE MICRO BREAK COMPARITORS FUNCTION
                ;       PROPERLY. THIS IS DONE BY HOLDING THE UPC AT 0 AND FLOATING
                ;       A ONE THRU THE MICRO BREAK REGISTER, THEN HOLDING THE UPC AT
                ;       ALL ONES AND FLOATING A ZERO THRU THE MICRO BREAK REGISTER THEN, ENSURING
                ;       THAT A MATCH OCCURS WHEN BOTH THE UPC AND BREAK REGISTER ARE AT
                ;       ZERO AND AT ONE.
                ;
                ;       THEN, THE UPC AND MICRO BREAK REGISTERS ARE SET TO ZERO AND A CHECK
                ;       IS MADE TO ENSURE THAT THE SYNC SIGNAL DOES NOT OCCURE DURING A
                ;       WRITE TO THE MICRO BREAK REGISTER.
                ;
                ;       SUBTST 1 - HOLD THE UPC AT 0 AND FLOAT A ONE THRU THE BREAK REG
                ;       SUBTST 2 - HOLD THE UPC AT -1 AND FLOAT A ZERO THRU THE BREAK REG
                ;       SUBTST 3 - CHECK THAT ALL 0'S CAUSES A MATCH AND THAT MATCH IS DISABLED
                ;                  WHEN THE CLOCK IS NOT IN CPT150.
                ;       SUBTST 4 - CHECK THAT ALL 1'S CAUSES A MATCH
                ;       SUBTST 5 - CHECK THAT MATCH IS DISABLED DURING A WRITE TO THE MICRO
                ;                  BREAK REGISTER.
                ;       SUBTST 6 - CHECK THAT THE CLOCK STOPS IN CPTO ON A MICRO BREAK MATCH
                ;                  AND THE SOMM BIT IS SET IN THE MCR REGISTER.
                ;
                ;       THE MATCH (OR NO MATCH) IS DETECTED BY TESTING THE OUTPUT OF THE
                ;       COMPARATORS ON THE V BUS.
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE MICRO BREAK COMPARATORS ON THE USC MODULE
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED V BUS CHANNEL, BIT, AND VALUE (MATCH SIGNAL)
                ;             RECEIVED V BUS CHANNEL, BIT, AND VALUE (MATCH SIGNAL)
                ;             LOOP COUNT - SUBTST 1 & 2 --INDICATES WHAT BIT POSITION THE
                ;               FLOATING ONE OR ZERO IS IN, I.E. 1=BIT 0, 2=BIT 1, ETC.
                ;
                ;       SUBTST 6 HAS THE FOLLOWING DATA TYPEOUT:
                ;                DATA: EXPECTED CONTENTS OF MCR OR V BUS CTRL REGISTER
                ;                      RECIEVED CONTENTS OF MCR OF V BUS CTRL REG
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC61--DATA IS STABLE ON THE COMPARATORS
                ;       SUBTST 2 - SYNC62--DATA IS STABLE ON THE COMPARATORS
                ;       SUBTST 3 - SYNC63--DATA IS STABLE ON THE COMPARATORS
                ;       SUBTST 4 - SYNC64--DATA IS STABLE ON THE COMPARATORS
                ;       SUBTST 5 - SYNC65--DATA IS STABLE ON THE COMPARATORS
                ;       SUBTST 6 - SYNC9B--CLOCK SHOULD STOP
                ;--
                ;*****************************************************************************
    000402      T27:
4115 000406             INITIALIZE
4116
4117 000410             SUBTEST
                ;/////////////////////////////////////////////////////////////////////////////
```

```
        000410  T27S1:
   4118         ;+
   4119         ; FIRST FLOAT A ONE THRU THE BREAK WITH THE UPC AT ZERO
   4120         ;-
   4121
   4122 000412          LOOP    I,1,13          ; CHECK 13 COMBINATIONS ON COMPARITORS
   4123 000424          INITIALIZE
   4124 000426          FLTONE  T22L1,I         ; GENERATE THE TEST PATTERN
   4125 000434          LDIDREG USCBRK,T22L1    ; PUT THE PATTERN IN THE BREAK REG
   4126 000442          ERLOOP
   4127 000444          INITIALIZE
   4128 000446          LOADCA  CONMCR,TMP47    ; SET MAINTENANCE RETURN BIT
   4129 000456          LDIDREG USCSTK,T22L2    ; PUT ADDRESS 0 ON THE STACK
   4130 000464          CLOCK   3               ; CLOCK TO CPT150
   4131 000470  SYNC61: TSTVB   TMP45           ; CHECK FOR NO MATCH
   4132 000476          IFERROR 42,USC14        ; MICRO BREAK COMPARITORS FAILED
   4133 000504          ENDLOOP I               ; CONTINUE
   4134
   4135 000510          SUBTEST
        ;////////////////////////////////////////////////////////////////////////////////
        000510  T27S2:
   4136         ;+
   4137         ; NOW FLOAT A ZERO WITH THE UPC AT ALL 1'S
   4138         ;-
   4139
   4140 000512          LOOP    I,1,13
   4141 000524          INITIALIZE
   4142 000526          FLTZRO  T22L1,I         ; GENERATE THE TEST PATTERN
   4143 000534          LDIDREG USCBRK,T22L1    ; PUT PATTERN IN BREAK REGISTER
   4144 000542          ERLOOP
   4145 000544          INITIALIZE
   4146 000546          LOADCA  CONMCR,TMP47    ; SET MAINTENANCE RETURN BIT
   4147 000556          LDIDREG USCSTK,T22M1    ; PUT ADDRESS ON STACK
   4148 000564          CLOCK   3               ; ADVANCE TO CPT150
   4149 000570  SYNC62: TSTVB   TMP45           ; CHECK FOR NO MATCH
   4150 000576          IFERROR 43,USC14        ; MICRO BREAK REGISTER FAILED
   4151 000604          ENDLOOP I               ; CONTINUE
   4152
   4153 000610          SUBTEST
        ;////////////////////////////////////////////////////////////////////////////////
        000610  T27S3:
   4154         ;+
   4155         ; NOW CHECK THAT ALL 0'S CAUSES A MATCH
   4156         ;-
   4157
   4158 000612          INITIALIZE
   4159 000614          LDIDREG USCBRK,T22L2    ; LOAD BREAK REG WITH ALL 0'S
   4160 000622          ERLOOP
   4161 000624          INITIALIZE
   4162 000626          LOADCA  CONMCR,TMP47    ; SET MAINTENANCE RETURN BIT
   4163 000636          LDIDREG USCSTK,T22L2    ; PUT ADDRESS ON STACK
   4164 000644          CLOCK   3               ; ADVANCE TO CPT150
   4165 000650  SYNC63: TSTVB   TMP46           ; CHECK FOR A MATCH
   4166 000656          CLOCK   1               ; GO TO CPTO
   4167 000662          IFERROR 44,USC14        ; MICRO BREAK COMPARITORS FAILED
   4168
   4169 000670          TSTVB   TMP45           ; CHECK THAT MATCH IS DISABLED IN CPTO
```

```
4170 000676            IFERROR ,USC14          ; MATCH DID NOT DISABLE IN CPTO
4171
4172 000704            SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
      000704    T27S4:
4173              ;+
4174              ; NOW CHECK THAT ALL 1'S WILL CAUSE A MATCH
4175              ;-
4176
4177 000706            LDIDREG USCBRK,T22M1    ; PUT ALL 1'S IN BREAK REG
4178 000714            ERLOOP
4179 000716            INITIALIZE
4180 000720            LOADCA  CONMCR,TMP47    ; SET MAINTENANCE RETURN BIT
4181 000730            LDIDREG USCSTK,T22M1    ; PUT ADDRESS ON STACK
4182 000736            CLOCK   3               ; ADVANCE TO CPT150
4183 000742  SYNC64:   TSTVB   TMP46           ; CHECK FOR A MATCH
4184 000750            IFERROR 45,USC14        ; MICRO BREAK COMPARITORS FAILED
4185
4186 000756            SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
      000756    T27S5:
4187              ;+
4188              ; NOW CHECK THAT THERE IS NO MATCH DURING A WRITE TO THE MICRO BREAK REG
4189              ;-
4190
4191 000760            ERLOOP
4192 000762            INITIALIZE
4193 000764            LOADCA  CONMCR,TMP47    ; SET MAINTENANCE RTN BIT
4194 000774            LDIDREG USCSTK,T22M1    ; PUT ADDRESS ON THE STACK
4195 001002            LOADCA  IDCS,T22L3      ; SETUP TO WRITE THE MICRO BREAK
4196 001012            LOADCA  TOIDLO,T22M1    ; SETUP DATA TO WRITE THE BREAK
4197 001022            CLOCK   3               ; GO TO CPT150
4198 001026  SYNC65:   TSTVB   TMP45           ; CHECK FOR NO MATCH
4199 001034            IFERROR ,USC14          ; ID WRITE TO UBREAK DID NOT INHIBIT MATCH
4200
4201 001042            SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
      001042    T27S6:
4202              ;+
4203              ; NOW CHECK THAT THE CLOCK STOPS IN CPTO WITH A MICRO BREAK MATCH AND THE SOMM
4204              ; BIT SET IN THE MCR REGISTER.
4205              ;-
4206
4207 001044            ERLOOP
4208 001046            INITIALIZE
4209 001050            LDIDREG USCSTK,T22L4    ; WRITE ADDRESS ON STACK TO BREAK ON
4210 001056            LOADCA  CONMCR,T22L5    ; SET THE MNTRTN BIT IN THE CONSOLE
4211 001066            LDIDREG USCBRK,T22L4    ; SET ADDRESS IN THE BRK REGISTER
4212 001074  SYNC9B:   LOADCA  CONMCR,T22L6    ; START THE CLOCK
4213 001104            CMPCA   EQ,CONMCR,T22L7 ; CHECK FOR CLOCKED STOP BIT
4214 001116            IFERROR ,USC14A         ; CLOCK DID NOT STOP
4215 001124            CMPCAM  EQ,VBCTRL,T22L9,,T22L9 ; CHECK FOR CPTO
4216 001140            IFERROR ,USC14A         ; CLOCK DID NOT STOP IN CPTO
4217 001146            LOADCA  CONMCR,T22L8    ; SET THE SBC BIT
4218 001156            SKIP
4219
4220 001162  USC14A:   REPORT  <CLK,USC>
```

```
                                ;-------------------------------------------------------------------------------
4225 001206  TMP45:  .WORD   1
4226 001210          VBUSG   0,116,0          ; UBREAK MATCH=0
4227 001212  TMP46:  .WORD   1
4228 001214          VBUSG   0,116,1          ; UBREAK MATCH=1
4229 001216  TMP47:  .WORD   MNTRTN+CLRUWRD+SBC
4230 001220          .WORD   CLRUWRD+SBC
4231 001222  T22L1:  .WORD   0,0
4232 001226  T22L2:  .WORD   0,0,0,0,1,0
4233 001242  T22L3:  .WORD   IDMAINT+IDWRITE+USCBRK
4234 001244  T22M1:  .WORD   17777
4235 001246  T22L4:  .WORD   0
4236 001250  T22L5:  .WORD   MNTRTN+CLRUWRD+SOMM+SBC
4237 001252  T22L6:  .WORD   CLRUWRD+SOMM+PROCEED
4238 001254  T22L7:  .WORD   CLRUWRD+SOMM+CLKSTPD
4239 001256  T22L8:  .WORD   CLRUWRD+SBC
4240 001260  T22L9:  .WORD   CCPTO
4241 001262          ENDDAT
                                ;-------------------------------------------------------------------------------
4242
```

```
4272                .SBTTL  T28     MICRO ECO
                    ;;*****************************************************************************
                    ;++
                    ;TEST   28      MICRO ECO
                    ;
                    ; TEST DESCRIPTION
                    ;       THIS TEST CHECKS THAT THE MICRO ECO CAUSES THE ECO ADDRESS
                    ;       TO BE CLOCKED INTO THE UPCSV REG.
                    ;       THIS IS DONE BY EXECUTING THE ECO'D ADDRESSES, CHECKING THAT THE
                    ;       ECO BIT, CLR UWORD, AND ABORT CYCLE ASSERT ON THE V BUS, AND
                    ;       CHECKING THAT THE ECO ADDRESS IS CLOCKED INTO THE PC SAVE REG.
                    ;
                    ;       SUBTST 1 - CHECK THAT THE V BUS SIGNALS ASSERT
                    ;       SUBTST 2 - CHECK THAT CORRECT ADDRESS IS CLOCKED INTO THE PC SAVE
                    ;
                    ; LOGIC DESCRIPTION
                    ;       THIS TEST CHECKS THE ADDRESS LINES GOING TO THE ECO FPLA AND THE
                    ;       DATA LINES COMING FROM THE FPLA, AND THE ECO ADDRESS LATCH, AND THAT
                    ;       THE U MULTIPLEXOR SWITCHES TO THE ECO LATCH ON THE USC MODULE.
                    ;
                    ; ERROR DESCRIPTION
                    ;       DATA: EXPECTED V BUS CHANNEL, BIT, AND VALUE OR ECO ADDRESS
                    ;             RECEIVED V BUS CHANNEL, BIT, AND VALUE OR ECO ADDRESS
                    ;
                    ;       NOTE: SUBTEST 1 PRINTS THE V BUS INFORMATION THAT FAILED
                    ;             SUBTEST 2 PRINTS THE ECO ADDRESS
                    ;
                    ; SYNC POINT DESCRIPTION
                    ;       SUBTST 1 - SYNC66--ECO ADDRESS IS STABLE AND LATCH IS LOADED.
                    ;       SUBTST 2 - SYNC67--ECO ADDRESS IS GATED THRU THE U MUX
                    ;--
                    ;;*****************************************************************************
    001262          T28:
4276 001266                 INITIALIZE
4277
4278 001270                 SUBTEST
                    ;//////////////////////////////////////////////////////////////////////////////
    001270          T28S1:
4279                ;+
4280                ; FIRST CHECK THAT ECO 06 ASSERTS AND THAT THE OTHER SIGNALS ASSERT
4281                ;-
4282
4283 001272                 LOOP    I,1,2
4284 001304                 ERLOOP
4285 001306                 FETCH   TMP50,I,NONOP   ; GET ECO ADDRESS
4286 001316          SYNC66: TSTVB   TMP51           ; CHECK THAT ECO BIT ASCERTED
4287 001324                 IFERROR 46,USC15        ; ECO DISPATCH BIT DID NOT ASCERT
4288 001332                 ENDLOOP I
4289
4290 001336                 SUBTEST
                    ;//////////////////////////////////////////////////////////////////////////////
    001336          T28S2:
4291                ;+
4292                ; NOW CHECK THAT ECO ADDRESS GETS STEARED THRU UMX INTO UPCSV REG
4293                ;-
4294
4295 001340                 LOOP    I,1,2
```

```
 4296 001352            ERLOOP
 4297 001354            INITIALIZE
 4298 001356            LOADCA  CONMCR,T23L1    ; LOAD MAINTENANCE RETURN BIT
 4299 001366            LDIDREG USCSTK,TMP50,I  ; PUT ECO ADDRESS ON STACK
 4300 001374            CLOCK   4               ; LATCH THE ECO ADDRESS
 4301 001400            LOADCA  CONMCR,T23L2    ; CLEAR ROM NOP
 4302 001410            CLOCK   1               ; GO TO CPT50
 4303 001414  SYNC67:   CLOCK   3               ; GO TO CPT0
 4304 001420            CMPPCSV TMP52,I         ; CHECK FOR THE CORRECT ADDRESS
 4305 001426            IFERROR 47,USC15        ; ECO ADDRESS DID NOT GET TO UPCSV REG
 4306 001434            ENDLOOP I
 4307 001440            SKIP
 4308
 4309 001444  USC15:    REPORT  <USC>           ;
 4310
 4311 001454            TESTDAT
                        ;-------------------------------------------------------------------------------
 4312 001460  TMP50:    .WORD   6525,11252      ; ADDRESS THAT GETS ECO'D
 4313                   ;   HEXDATA     0D55, 12AA
 4314 001464  TMP51:    .WORD   3
 4315 001466            VBUSG   0,17,1          ; ECO DISPATCH 06 BIT
 4316 001470            VBUSG   1,10,1          ; CEHF CLR UWORD (1) H
 4317 001472            VBUSG   3,14,1          ; USCB ABORT CYCLE
 4318 001474  TMP52:    .WORD   10552,10525     ; ECO ADDRESS
 4319                   ;   HEXDATA      11AA, 1155
 4320 001500  T23L1:    .WORD   MNTRTN+CLRUWRD+SBC
 4321 001502  T23L2:    .WORD   SBC
 4322 001504            ENDDAT
                        ;-------------------------------------------------------------------------------
 4323
 4361
```

```
                .SBTTL  T29     MICRO STACK PUSH WITH CS PARITY ERROR
                ;;*****************************************************************************
                ;++
                ;TEST   29      MICRO STACK PUSH WITH CS PARITY ERROR
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THAT THE CS PARITY ERROR UTRAP INHIBITS THE
                ;       UPC SAVE REGISTER FROM BEING CLOCKED, THAT THE UTRAP SIGNAL IS
                ;       ASSERTED, THAT THE CORRECT TRAP VECTOR IS ASSERTED, AND THAT THE
                ;       "FORCE UPC 12" BIT FUNCTIONS PROPERLY, AND THAT THE "CANCEL" SIGNAL
                ;       ON THE TBM MODULE ASSERTS.
                ;
                ;       SUBTST 1 - CHECK THAT THE UTRAP SIGNAL AND CS PARITY ERROR SIGNAL ASSERT,
                ;                  AND "CANCEL" SIGNAL ASSERT, AND THAT "CLR UTRAP" IS ASSERTED
                ;                  AT CPT100 OF THE MICRO STACK PUSH.
                ;       SUBTST 2 - CHECK THAT THE UPC REGISTER DOES NOT CLOCK AND THAT THE
                ;                  STACK GETS PUSHED.
                ;       SUBTST 3 - CHECK THAT THE TRAP VECTOR IS CORRECT AND THAT THE FORCE
                ;                  UPC 12 BIT FUNCTIONS PROPERLY.
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE LOGIC ON THE CEH MODULE THAT GENERATES THE
                ;       THE UTRAP SIGNAL AND PARITY ERROR SIGNAL, THE LOGIC ON THE ICL MODULE
                ;       THAT GENERATES THE MICRO TRAP ADDRESS, THE LOGIC ON THE USC MODULE
                ;       THAT INHIBITS THE PC SAVE CLOCK DURING A CS PARITY ERROR, AND
                ;       THE LOGIC ON THE USC MODULE THAT STEERS THE U MUX AND THE MULTIPLEXORS
                ;       FOR BRANCH ENABLE 10.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED V BUS CHANNEL, BIT, AND VALUE OR PC SAVE REG
                ;             RECEIVED V BUS CHANNEL, BIT, AND VALUE OR PC SAVE REG
                ;             LOOP COUNT - SUBTST 3 ONLY -- INDICATES THE EXPECTED VALUE OF
                ;               THE UPC 12 ADDRESS LINE, I.E. 1=DEASSERTED, 2=ASSERTED
                ;
                ;       NOTE: SUBTEST 1 PRINT THE V BUS INFORMATION THAT FAILED
                ;             SUBTESTS 2 & 3 PRINT THE CONTENTS OF THE PC SAVE REG
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC68--UTRAP AND CS PARITY ERROR SIGNALS ASSERTED
                ;       SUBTST 2 - SYNC69--PC SAVE CLOCK INHIBIT ACTIVE AND USTACK DEC ASSERTED
                ;       SUBTST 3 - SYNC6A--TRAP ADDRESS ACTIVE ON THE U MUX
                ;--
                ;;*****************************************************************************
    001504      T29:
4369 001510             INITIALIZE
4370
4371 001512             SUBTEST
                ;//////////////////////////////////////////////////////////////////////////////
    001512      T29S1:
4372            ;+
4373            ; FIRST CHECK THAT UTRAP AND PARITY ERROR ASSERT OVER THE V BUS
4374            ;-
4375
4376 001514             BLKMIC  T24L3,,10417,1  ; LOAD BRANCH . IN PARITY VECTOR
4377 001530             BLKMIC  XTMP52,,110000,1 ; LOAD MICRO INSTR WITH BAD PARITY
4378 001544             ERLOOP
4379 001546             INITIALIZE
```

```
4380 001550          FETCH   10000,,NONOP    ; EXECUTE ADDRESS 0
4381 001560          CLOCK   4
4382 001564  SYNC68: TSTVB   TMP201          ; CHECK FOR MICRO TRAP AND PARITY ERROR
4383 001572          IFERROR 136,1$          ; EITHER UTRAP OR PARITY ERROR SIGNAL FROM
4384                                         ; CEH DID NOT ASCERT
4385 001600          CLOCK   1               ; GO TO CPT75
4386 001604          TSTVB   T24L1           ; CHECK THAT "CLR UTRAP" ASSERTS
4387 001612          CLOCK   3               ; GO BACK TO CPTO
4388 001616          IFERROR ,USC16          ; CLR UTRAP DID NOT ASSERT
4389 001624          TSTVB   T24L4           ; CHECK THAT "CS PE TRAP" DEASCERTS
4390 001632          IFERROR ,5$             ; CALL OUT CEH
4391 001640          SKIP    SUBTST          ; CONTINUE
4392
4393 001644  1$:     TSTVB   T28V1           ; CHECK "CS PE TRAP" ASCERTED
4394 001652          CHKPNT  2$              ; BRANCH IF OK
4395 001660  5$:     REPORT  <CEH>
4396 001670  2$:     TSTVB   T28V2           ; CHECK UTRAP AND ABORT CYCLE ASCERTED
4397 001676          CHKPNT  3$              ; BRANCH IF OK
4398 001704          REPORT  <CEH,USC>
4399 001714  3$:     TSTVB   T28V3           ; CHECK CLR UWORD ASCERTED
4400 001722          CHKPNT  4$              ; BRANCH IF OK
4401 001730          REPORT  <USC,CEH>
4402 001740  4$:     REPORT  <TBM>           ; MUST HAVE BEEN CANCEL THAT DID NOT ASCERT
4403
4404 001750          SUBTEST
             ;////////////////////////////////////////////////////////////////////////////////
     001750  T29S2:
4405                 ;+
4406                 ; CHECK THAT THE PC SAVE REGISTER DOES NOT CLOCK AND THAT THE MICRO STACK
4407                 ; IS PUSHED WITH THE ADDRESS OF THE PARITY ERROR
4408                 ;-
4409
4410 001752          ERLOOP
4411 001754          INITIALIZE
4412 001756          LDIDREG USCSTK,T24M1    ; PUT KNOWN DATA ON THE STACK
4413 001764          FETCH   10000,,NONOP    ; EXECUTE WORD WITH BAD PARITY
4414 001774          CLOCK   4               ; ASSERT UTRAP
4415 002000  SYNC69: CLOCK   4               ; PUSH THE U STACK
4416 002004          READID  USCSTK          ; GET THE TOP OF THE STACK
4417 002010          CMPCA   EQ,IDREGLO,T24L2 ; SEE IF ADDRESS 0 WAS PUSHED
4418 002022          IFERROR 137,USC16       ; CS PE TRAP DID NOT INHIBIT UPCSV CLOCK
4419 002030          READID  USCSTK          ; DID THE STACK DECREMENT?
4420 002034          CMPCA   EQ,IDREGLO,T24M1        ;
4421 002046          IFERROR 140,USC16       ; STACK DECREMENT DID NOT ASCERT ON UTRAP
4422
4423 002054          SUBTEST
             ;////////////////////////////////////////////////////////////////////////////////
     002054  T29S3:
4424                 ;+
4425                 ; NOW CHECK THAT THE FORCE UPC12 BIT IN THE CONSOLE CAUSES THE UTRAP
4426                 ; ADDRESS TO GO TO THE WCS ADDRESS BANK
4427                 ; ALSO CHECKS THAT "UTRAP SEQ" CLEARS THE UJMP FIELD ON THE SEQ.
4428                 ;-
4429
4430 002056          LOOP    I,1,2
4431 002070          ERLOOP
4432 002072          INITIALIZE
```

```
4433 002074            FETCH   10000,,NONOP     ; GET UWORD WITH BAD PARITY
4434 002104            LOADCA  CONMCR,TMPX52,I  ; SET OR CLEAR UPC12 BIT IN CONSOLE
4435 002114            CLOCK   4                ; ASSERT UTRAP
4436 002120            CLOCK   1                ; ASSERT TRAP ADDRESS THRU U MUX
4437 002124  SYNC6A:   CLOCK   3                ; CLOCK THE PC SAVE
4438 002130            CMPPCSV X52,I            ; CHECK THAT CORRECT ADDRESS GOT LOADED
4439 002136            IFERROR ,USC16           ; UPC BIT 12 FAILED
4440 002144            ENDLOOP I
4441 002150            SKIP
4442
4443 002154  USC16:    REPORT  <USC>
4444
4445 002164            TESTDAT
                       ;-------------------------------------------------------------------------------
4446 002170  TMP201:   .WORD   5
4447 002172            VBUSG   0,16,1           ; USCB UTRAP H
4448 002174            VBUSG   1,10,1           ; CLR UWORD H
4449 002176            VBUSG   3,14,1           ; ABORT CYCLE H
4450 002200            VBUSG   1,73,1           ; CEH CS PARITY ERROR
4451 002202            VBUSG   3,3,0            ; TBMU CANCEL L
4452 002204  TMPX52:   .WORD   SBC
4453 002206            .WORD   SBC+UPC12
4454 002210  XTMP52:   .WORD   11777
4455 002212            .WORD   0
4456 002214            .WORD   0
4457 002216            .WORD   0
4458 002220            .WORD   0
4459 002222            .WORD   0
4460 002224  X52:      .WORD   417,10417        ; CS PARITY ERROR VECTOR
4461                   ;   HEXDATA     10F, 110F
4462 002230  T24M1:    .WORD   17777
4463                   ;   HEXDATA     1FFF
4464 002232  T24L1:    .WORD   1
4465 002234            VBUSG   1,57,0           ; CLR UTRAP L
4466 002236  T24L2:    .WORD   10000
4467 002240  T24L3:    .WORD   10417,0,0,0,0,0
4468 002254  T24L4:    .WORD   1
4469 002256            VBUSG   1,73,0           ; CS PE TRAP H
4470
4471 002260  T28V1:    .WORD   1
4472 002262            VBUSG   1,73,1           ; CS PE TRAP H
4473 002264  T28V2:    .WORD   2
4474 002266            VBUSG   0,16,1           ; USCB UTRAP H
4475 002270            VBUSG   3,14,1           ; ABORT CYCLE H
4476 002272  T28V3:    .WORD   1
4477 002274            VBUSG   1,10,1           ; CLR UWORD (1) H
4478 002276            ENDDAT
                       ;-------------------------------------------------------------------------------
4479
```

```
                        .SBTTL  SECTION NUMBER 0A
                        .PSECT  OVR12
                        .SBTTL  T2A     INITIALIZE MICRO TRAP
                        ;;*****************************************************************************
                        ;++
                        ;TEST   2A      INITIALIZE MICRO TRAP
                        ;
                        ; TEST DESCRIPTION
                        ;       THIS TEST CHECKS THAT THE SYSTEM INIT CAUSES THE UJMP FIELD ON THE
                        ;       MICRO SEQUENCER TO BE CLEARED AND THAT THE MICRO TRAP ADDRESS IS
                        ;       GATED ONTO THE UPC LINES FROM THE ICL MODULE.
                        ;
                        ; LOGIC DESCRIPTION
                        ;       THIS TEST CHECKS THE LOGIC ON THE MICRO SEQUENCER THAT CLEARS THE
                        ;       UJMP FIELD LATCHES AND THE LOGIC ON THE ICL MODULE THAT GENERATES
                        ;       THE MICRO TRAP ADDRESS, AND THE LOGIC ON THE MICRO SEQUENCER
                        ;       THAT ENABLES THE TRAP VECTOR ONTO THE NUA BUS.
                        ;
                        ; ERROR DESCRIPTION
                        ;       DATA: EXPECTED CONTENTS OF THE UPC SAVE REG
                        ;             RECEIVED CONTENTS OF THE UPC SAVE REG
                        ;
                        ; SYNC POINT DESCRIPTION
                        ;       ADDRESS SYNCAF--MICRO TRAP VECTOR ENABLED ONTO NUA
                        ;--
                        ;;*****************************************************************************
    000034              T2A:
4505 000040                     INITIALIZE
4506
4507 000042                     BLKMIC  T39L1,,11777,1  ; LOAD MICRO INSTRUCTION TO LOAD JMP FIELD
4508 000056                     ERLOOP
4509 000060                     FETCH   11777,,NONOP    ; GET THE INSTRUCTION
4510 000070                     LOADCA  CONMCR,T39L2    ; SET THE SYSTEM INIT BIT
4511 000100                     LOADCA  CONMCR,T39L3    ; CLEAR SYSTEM INIT
4512 000110                     CLOCK   2               ; PUT TRAP VECTOR ON NUA BUS
4513 000114                     TSTVB   T39L5           ; CHECK VECTOR BITS ON V BUS
4514 000122                     IFERROR ,USC39A         ; CALL OUT CEH
4515 000130             SYNCAF: CLOCK   2               ; LATCH VECTOR INTO UPC SAVE
4516 000134                     CMPPCSV T39L4           ; CHECK THAT VECTOR IS CORRECT
4517 000142                     IFERROR ,USC39          ; MICRO TRAP VECTOR INCORRECT
4518 000150                     SKIP
4519
4520 000154             USC39:  REPORT  <USC>           ;
4521 000164             USC39A: REPORT  <CEH>
4522
4523 000174                     TESTDAT
                        ;-----------------------------------------------------------------------------
4524 000200             T39L1:
4525                    ;*13FF: BEN/0,J/13FF
4526 000200                     .WORD   11777
4527 000202                     .WORD   0
4528 000204                     .WORD   4000
4529 000206                     .WORD   600
4530 000210                     .WORD   74
4531 000212                     .WORD   0
4532
4533 000214             T39L2:  .WORD   INIT+SBC
```

```
 4534 000216  T39L3:  .WORD   SBC
 4535 000220  T39L4:  .WORD   400
 4536         ;   HEXDATA     100
 4537 000222  T39L5:  .WORD   4
 4538 000224          VBUSG   0,55,0          ; UTRAP VECTOR 0 H
 4539 000226          VBUSG   0,61,0          ; UTRAP VECTOR 1 H
 4540 000230          VBUSG   0,65,0          ; UTRAP VECTOR 2 H
 4541 000232          VBUSG   0,71,0          ; UTRAP VECTOR 3 H
 4542 000234          ENDDAT
              ;-------------------------------------------------------------------------------
 4543
 4544 000234          FORCEOVERLAY
              .SBTTL  SECTION NUMBER 0B
      000000          .PSECT  OVR13
 4545
```

```
                        .SBTTL  T2B     ID BUS TO DATA PATH INTERFACE
                        ;;*****************************************************************************
                        ;++
                        ;TEST   2B      ID BUS TO DATA PATH INTERFACE
                        ;
                        ; TEST DESCRIPTION
                        ;       THIS TEST CHECKS THE DATA PATH FROM THE ID BUS TO THE Q REGISTER,
                        ;       FROM THE Q REG TO THE D REGISTER (THRU THE DAL), AND FROM THE
                        ;       D REG BACK TO THE ID BUS, WITH A FLOATING ONE AND ZERO PATTERN.
                        ;
                        ;       SUBTST 1 - CHECK WITH A FLOATING ZERO PATTERN
                        ;       SUBTST 2 - CHECK WITH A FLOATING ONE PATTERN
                        ;
                        ; LOGIC DESCRIPTION
                        ;       THIS TEST CHECKS THE LOGIC ON THE DAP MODULE THAT GENERATES THE
                        ;       LOAD SIGNALS FOR THE Q AND D REGISTER, THE LOGIC ON THE DBP, DCP
                        ;       DDP, AND DEP MODULES THAT MAKEUP THE DATA PATH FOR THE D AND Q REGISTER,
                        ;       AND THE LOGIC ON THE ICL MODULE THAT GENERATES THE ID TO Q AND
                        ;       D TO ID SIGNALS WHEN THE ID BUS IS IN MAINTENANCE MODE.
                        ;
                        ; ERROR DESCRIPTION
                        ;       DATA: EXPECTED CONTENTS OF THE D REGISTER
                        ;             RECEIVED CONTENTS OF THE D REGISTER
                        ;             LOOP COUNT -- INDICATES THE BIT POSITION OF THE FLOATING ONE
                        ;               OR ZERO PATTERN, I.E. 1=BIT 0, 2=BIT 1, 3=BIT 2, ETC.
                        ;
                        ; SYNC POINT DESCRIPTION
                        ;       SUBTST 1 - SYNC6C--Q REGISTER IS LOADED FROM THE ID BUS
                        ;                  SYNC6D--D REGISTER IS LOADED FROM THE Q REGISTER
                        ;                  SYNC6E--ID BUS IS DRIVEN FROM THE D REGISTER
                        ;       SUBTST 2 - SYNC6F--Q REGISTER IS LOADED FROM THE ID BUS
                        ;                  SYNC70--D REGISTER IS LOADED FROM THE Q REGISTER
                        ;                  SYNC71--ID BUS IS DRIVEN FROM THE D REGISTER
                        ;--
                        ;;*****************************************************************************
 000034                 T2B:
4582 000040                     INITIALIZE
4583 000042                     BLKMIC  TMP53,,10000,2  ; LOAD THE MICRO INSTRUCTIONS
4584
4585 000056                     SUBTEST
                        ;//////////////////////////////////////////////////////////////////////////////
 000056                 T2BS1:
4586                    ;+
4587                    ; FIRST FLOAT A ZERO THRU THE D AND Q REGISTER
4588                    ;-
4589
4590 000060                     LOOP    I,1,32
4591 000072                     ERLOOP
4592 000074                     FLTZRO  TMP55,I         ; GENERATE THE TEST PATTERN
4593 000102                     INITIALIZE
4594 000104                     LOADCA  TOIDLO,TMP55    ; LOAD THE LOW 16 BITS OF THE PATTERN
4595 000114                     LOADCA  TOIDHI,TMP55+2  ; LOAD THE HIGH 16 BITS
4596 000124                     LOADCA  IDCS,TMP55A     ; SETUP TO WRITE THE Q REGISTER
4597 000134             SYNC6C: CLOCK   4               ; LOAD THE Q REGISTER
4598 000140                     FETCH   10000,,NONOP    ; GET THE MICRO INSTRUCTION
4599 000150             SYNC6D: CLOCK   4               ; EXECUTE THE MICROINSTRUCTION
4600 000154             SYNC6E: READID  RWDQ            ; READ THE D REG BACK
```

```
4601 000160          CMPCAD  EQ,IDREGLO,TMP55 ; CHECK THE RECEIVED DATA
4602 000204          IFERROR 102,1$          ; BIT STUCK IN ID BUS INTERFACE TO DATA
4603                                         ; PATH OR IN THE Q-D TRANSFER PATH.
4604 000212          ENDLOOP I               ; CONTINUE
4605 000216          SKIP    SUBTEST
4606
4607 000222  1$:     INITIALIZE
4608 000224          LOADCA  TOIDLO,TMP55    ; SETUP TO WRITE THE Q REGISTER IN
4609 000234          LOADCA  TOIDHI,TMP55+2  ; SINGLE TIME STATE
4610 000244          LOADCA  IDCS,TMP55A     ; ...
4611 000254          CLOCK   2               ; START THE WRITE TO THE Q REGISTER
4612 000260          TSTVB   T2AV1           ; CHECK "ID TO Q" ACTIVE
4613 000266          CHKPNT  2$              ; BRANCH IF OK
4614 000274          REPORT  <ICL>           ; "ID TO Q" NOT ACTIVE
4615 000304  2$:     CLOCK   2               ; FINISH ID BUS WRITE
4616 000310          FETCH   10000,,NONOP    ; GET MICRO INSTR TO XFR Q TO D
4617 000320          CLOCK   4               ; EXECUTE IT
4618 000324          LOADCA  IDCS,T2AL1      ; SETUP TO READ D REGISTER IN SINGLE STATE
4619 000334          CLOCK   2               ; START THE READ
4620 000340          TSTVB   T2AV2           ; CHECK "D TO ID" ACTIVE
4621 000346          CHKPNT  DP1             ; BRANCH IF OK
4622 000354          REPORT  <ICL>           ; "D TO ID" NOT ACTIVE
4623
4624          ;+
4625          ; NOW EXAMINE THE EXPECTED AND RECEIVED DATA TO DETERMINE WHICH
4626          ; BYTE FAILED
4627          ;-
4628
4629 000364  DP1:     MASK    TMP55,MBYTE0    ; ONLY WANT BYTE 0 OF EXPECTED DATA
4630 000372          CMPCAM  ,IDREGLO,MBYTE0,,TMP55 ; CHECK RECEIVED DATA BYTE 0
4631 000406          CHKPNT  4$              ; BRANCH IF OK TO 4$
4632 000414          FLTZRO  TMP55,I         ; REGENERATE TEST DATA
4633 000422          MASK    TMP55,MBYTE1    ; ONLY WANT BYTE1
4634 000430          CMPCAM  ,IDREGLO,MBYTE1,,TMP55 ; CHECK IF BYTE 1 IS BAD
4635 000444          CHKPNT  5$              ; GO TO 5$ IF IT IS OK
4636 000452          CMPCA   ,IDREGHI,TMP55+2 ; IS UPPER WORD GOOD?
4637 000464          CHKPNT  6$              ; BRANCH IF YES
4638 000472          REPORT  <DAP,DBP>       ; ALL  BYTES ARE BAD
4639 000502  6$:     REPORT  <DBP,DCP,DDP>   ; BYTES 0 AND 1 ARE BAD
4640 000512  5$:     REPORT  <DBP,DCP>       ; BYTE 0 BAD, BYTE1 GOOD
4641
4642 000522  4$:     FLTZRO  TMP55,I         ; REGENERATE TEST PATTERN
4643 000530          MASK    TMP55,MBYTE1    ; ONLY WANT BYTE 1
4644 000536          CMPCAM  ,IDREGLO,MBYTE1,,TMP55 ; IS BYTE 1 BAD?
4645 000552          CHKPNT  7$              ; BRANCH IF NO
4646 000560          REPORT  <DBP,DDP>       ; BYTE 0 GOOD, BYTE 1 BAD
4647
4648 000570  7$:     CMPCA   ,IDREGHI,TMP55+2 ; ARE BYTES 2,3 GOOD?
4649 000602          CHKPNT  8$              ; BRANCH IF YES
4650 000610          REPORT  <DBP,DEP>       ; BYTES 0,1 GOOD, BYTES 2,3 BAD
4651 000620  8$:     REPORT  <DAP,DBP,DCP,DDP,DEP> ; ALL BYTES GOOD (SHOULD NEVER HAPPEN)
4652
4653
4654 000630          SUBTEST
          ;////////////////////////////////////////////////////////////////////////////////
     000630  T2BS2:
4655          ;+
```

```
                        ; NOW FLOAT A ONE THRU THE DATA PATH
                        ;-

000632                          LOOP    I,1,32
000644                          ERLOOP
000646                          FLTONE  TMP55,I         ; GENERATE THE TEST PATTERN
000654                          INITIALIZE
000656                          LOADCA  TOIDLO,TMP55
000666                          LOADCA  TOIDHI,TMP55+2
000676                          LOADCA  IDCS,TMP55A     ; SETUP TO TO AN ID BUS WRITE
000706  SYNC6F:                 CLOCK   4               ; LOAD THE Q REGISTER
000712                          FETCH   10000,,NONOP    ; GET THE MICRO INSTRUCTION
000722  SYNC70:                 CLOCK   4               ; EXECUTE THE MICRO INSTRUCTION
000726  SYNC71:                 READID  RWDQ            ; READ THE D REGISTER
000732                          CMPCAD  EQ,IDREGLO,TMP55 ; CHECK THE RECEIVED DATA
000756                          IFERROR 103,DP1A        ; BIT STUCK IN DATA PATH
000764                          ENDLOOP I               ; CONTINUE
000770                          SKIP

                        ;+
                        ; FIND OUT WHICH BYTE IS BAD
                        ;-

000774  DP1A:                  MASK    TMP55,MBYTE0    ; ONLY WANT BYTE 0
001002                          CMPCAM  ,IDREGLO,MBYTE0,,TMP55 ; IS BYTE 0 BAD?
001016                          CHKPNT  4$              ; BRANCH IF NO
001024                          FLTONE  TMP55,I         ; REGENERATE TEST PATTERN
001032                          MASK    TMP55,MBYTE1    ; ONLY WANT BYTE 1
001040                          CMPCAM  ,IDREGLO,MBYTE1,,TMP55 ; IS BYTE 1 BAD?
001054                          CHKPNT  5$              ; BRANCH IF NO
001062                          CMPCA   ,IDREGHI,TMP55+2 ; ARE BYTES 2,3 BAD?
001074                          CHKPNT  6$              ; BRANCH IF NO
001102                          REPORT  <DAP,DBP>       ; ALL THE BYTES ARE BAD
001112  6$:                     REPORT  <DBP,DCP,DDP>   ; BYTES 0 AND 1 BAD, 2,3 ARE GOOD
001122  5$:                     REPORT  <DBP,DCP>       ; BYTE 0 BAD, BYTE 1 GOOD

001132  4$:                     FLTONE  TMP55,I         ; REGENERATE TEST PATTERN
001140                          MASK    TMP55,MBYTE1    ; ONLY WANT BYTE 1
001146                          CMPCAM  ,IDREGLO,MBYTE1,,TMP55 ; IS BYTE 1 BAD?
001162                          CHKPNT  7$              ; BRANCH IF NO
001170                          REPORT  <DBP,DDP>       ; BYTE 0 GOOD, BYTE 1 BAD
001200  7$:                     CMPCA   ,IDREGHI,TMP55+2 ; BYTES 2,3 BAD?
001212                          CHKPNT  8$              ; BRANCH IF NO
001220                          REPORT  <DBP,DEP>       ; BYTES 0,1 GOOD, BYTES 2,3 BAD
001230  8$:                     REPORT  <DAP,DBP,DCP,DDP,DEP> ; ALL BYTES GOOD (SHOULD NEVER HAPPEN)

001240                          TESTDAT
                        ;-------------------------------------------------------------------------------
001244  TMP53:
                        ;*1000: DK/Q,J/1001             ; D GETS Q MICRO INSTRUCTION
001244                          .WORD   10001
001246                          .WORD   0
001250                          .WORD   4000
001252                          .WORD   600
001254                          .WORD   000074
001256                          .WORD   6000
001260  TMP56:
```

```
4712                ;*1001: BEN/0,J/1001            ; NOP MICRO WORD
4713 001260                .WORD   10001
4714 001262                .WORD   0
4715 001264                .WORD   4000
4716 001266                .WORD   600
4717 001270                .WORD   000074
4718 001272                .WORD   0
4719 001274   TMP54:        .WORD   SBC               ; DATA TO LOAD INTO MCR REG
4720 001276   TMP55:        .WORD   0,0               ; WORKING LOCATION FOR DATA PATTERNS
4721 001302   TMP55A:       .WORD   IDMAINT+IDWRITE+RWDQ
4722 001304   T2AL1:        .WORD   IDMAINT+RWDQ      ; D REGISTER READ
4723 001306   MBYTE0:       .WORD   377               ; MASK FOR BYTE 0
4724 001310   MBYTE1:       .WORD   177400            ; MASK FOR BYTE 1
4725 001312   T2AV1:        .WORD   1
4726 001314                VBUSG   2,144,0           ; ID TO Q L
4727 001316   T2AV2:        .WORD   1
4728 001320                VBUSG   2,143,0           ; D TO ID L
4729 001322                ENDDAT
                    ;-------------------------------------------------------------------------------
4730
```

```
4760            .SBTTL  T2C     ALU ALEG DATA PATH ''RAMX_Q''
                ;;*****************************************************************************
                ;++
                ;TEST   2C      ALU ALEG DATA PATH ''RAMX_Q''
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST FLOATS A ZERO AND A ONE THRU THE ALEG OF THE ALU. THIS IS
                ;       DONE BY LOADING THE Q REGISTER, FROM THE ID BUS, WITH THE DATA PATTERN,
                ;       EXECUTING A MICRO INSTRUCTION THAT ROUTES THE PATTERN THRU THE RAMX,
                ;       TO THE AMX, THRU THE ALU ALEG, AND BACK TO THE D REGISTER. THE PATTERN
                ;       IS THEN READ FROM THE D REGISTER AND CHECKED.
                ;
                ;       SUBTST 1 - FLOAT A ZERO PATTERN
                ;       SUBTST 2 - FLOAT A ONE PATTERN
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE CONTROL LOGIC ON THE DAP MODULE AND THE DATA PATHS
                ;       ON THE DBP, DCP, DDP, AND DEP MODULES THAT ROUTE THE Q REGISTER THRU THE
                ;       RAMX, THE RAMX THRU THE AMX, THE ALU SELECT OF F=A, THE SHF SELECT,
                ;       AND THE DMUX SELECT, AND THE D REGISTER CLOCK CONTROL.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED CONTENT OF THE D REG AFTER THE Q IS ROUTED THRU THE ALU
                ;             RECEIVED CONTENTS OF THE D REG
                ;             LOOP COUNT -- INDICATES WHAT BIT POSITION THE FLOATING ONE OR
                ;               ZERO IS IN, I.E. 1=BIT 0, 2=BIT 1, 3=BIT 2, ETC.
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC72--DATA IS IN Q REG AND BEING ROUTED THRU ALU
                ;       SUBTST 2 - SYNC73--DATA IS IN Q REG AND BEING ROUTED THRU ALU.
                ;--
                ;;*****************************************************************************
    001322      T2C:
4764 001326             INITIALIZE
4765 001330             BLKMIC  TMP57,,10000,1  ; LOAD INSTRUCTION TO TRANSFER Q TO D
4766
4767 001344             SUBTEST
                ;//////////////////////////////////////////////////////////////////////////////
    001344      T2CS1:
4768            ;+
4769            ; FIRST FLOAT A ZERO THRU THE DATA PATH
4770            ;-
4771
4772 001346             LOOP    I,1,32
4773 001360             ERLOOP
4774 001362             FLTZRO  XX57,I          ; GENERATE THE TEST PATTERN
4775 001370             INITIALIZE
4776 001372             LDIDREG RWDQ,XX57       ; LOAD Q REG WITH DATA PATTERN
4777 001400             FETCH   10000,,NONOP    ; GET THE MICRO INSTRUCTION
4778 001410     SYNC72: CLOCK   4               ; EXECUTE THE MICRO INSTRUCTION
4779 001414             READID  RWDQ            ; READ THE D REGISTER
4780 001420             CMPCAD  EQ,IDREGLO,XX57 ; CHECK THE CONTENTS OF THE D REG
4781 001444             IFERROR 104,DP2         ; BIT STUCK IN ALU ALEG DATA PATH
4782 001452             ENDLOOP I               ; CONTINUE
4783 001456             SKIP    SUBTST
4784
4785
```

```
4786                    ;+
4787                    ; FIND OUT WHICH BYTE IS BAD
4788                    ;-
4789
4790 001462  DP2:      MASK    XX57,NBYTE0      ; ONLY WANT BYTE 0
4791 001470             CMPCAM  ,IDREGLO,NBYTE0,,XX57 ; IS BYTE 0 BAD?
4792 001504             CHKPNT  4$               ; BRANCH IF NO
4793 001512             FLTZRO  XX57,I           ; REGENERATE TEST PATTERN
4794 001520             MASK    XX57,NBYTE1      ; ONLY WANT BYTE 1
4795 001526             CMPCAM  ,IDREGLO,NBYTE1,,XX57 ; IS BYTE 1 BAD?
4796 001542             CHKPNT  5$               ; BRANCH IF NO
4797 001550             CMPCA   ,IDREGHI,XX57+2  ; ARE BYTES 2,3 BAD?
4798 001562             CHKPNT  6$               ; BRANCH IF NO
4799 001570             REPORT  <DAP,DBP>        ; ALL THE BYTES ARE BAD
4800 001600  6$:        REPORT  <DBP,DCP,DDP>    ; BYTES 0 AND 1 BAD, 2,3 ARE GOOD
4801 001610  5$:        REPORT  <DBP,DCP>        ; BYTE 0 BAD, BYTE 1 GOOD
4802
4803 001620  4$:        FLTZRO  XX57,I           ; REGENERATE TEST PATTERN
4804 001626             MASK    XX57,NBYTE1      ; ONLY WANT BYTE 1
4805 001634             CMPCAM  ,IDREGLO,NBYTE1,,XX57 ; IS BYTE 1 BAD?
4806 001650             CHKPNT  7$               ; BRANCH IF NO
4807 001656             REPORT  <DBP,DDP>        ; BYTE 0 GOOD, BYTE 1 BAD
4808 001666  7$:        CMPCA   ,IDREGHI,XX57+2  ; BYTES 2,3 BAD?
4809 001700             CHKPNT  8$               ; BRANCH IF NO
4810 001706             REPORT  <DBP,DEP>        ; BYTES 0,1 GOOD, BYTES 2,3 BAD
4811 001716  8$:        REPORT  <DAP,DBP,DCP,DDP,DEP> ; ALL BYTES GOOD (SHOULD NEVER HAPPEN)
4812
4813 001726             SUBTEST
                        ;;;
```

```
                        CMPCAM  ,IDREGLO,NBYTE1,,XX57 ; IS BYTE 1 BAD?
                        CHKPNT  5$                    ; BRANCH IF NO
                        CMPCA   ,IDREGHI,XX57+2 ; ARE BYTES 2,3 BAD?
                        CHKPNT  6$                    ; BRANCH IF NO
                        REPORT  <DAP,DBP>             ; ALL THE BYTES ARE BAD
                6$:     REPORT  <DBP,DCP,DDP>   ; BYTES 0 AND 1 BAD, 2,3 ARE GOOD
                5$:     REPORT  <DBP,DCP>             ; BYTE 0 BAD, BYTE 1 GOOD

                4$:     FLTONE  XX57,I                ; REGENERATE TEST PATTERN
                        MASK    XX57,NBYTE1           ; ONLY WANT BYTE 1
                        CMPCAM  ,IDREGLO,NBYTE1,,XX57 ; IS BYTE 1 BAD?
                        CHKPNT  7$                    ; BRANCH IF NO
                        REPORT  <DBP,DDP>             ; BYTE 0 GOOD, BYTE 1 BAD
                7$:     CMPCA   ,IDREGHI,XX57+2 ; BYTES 2,3 BAD?
                        CHKPNT  8$                    ; BRANCH IF NO
                        REPORT  <DBP,DEP>             ; BYTES 0,1 GOOD, BYTES 2,3 BAD
                8$:     REPORT  <DAP,DBP,DCP,DDP,DEP> ; ALL BYTES GOOD (SHOULD NEVER HAPPEN)


                        TESTDAT
                ;-------------------------------------------------------------------------------
                TMP57:
                ;*1000: RAMX/Q,AMX/RAMX,ALU/A,SHF/ALU,DK/SHF,J/1000 ; TRANSFER Q TO D THRU ALU
                        .WORD   10000
                        .WORD   0
                        .WORD   4000
                        .WORD   600
                        .WORD   020074
                        .WORD   4001
                X57:    .WORD   SBC
                XX57:   .WORD   0,0
                NBYTE0: .WORD   377
                NBYTE1: .WORD   177400
                        ENDDAT
                ;-------------------------------------------------------------------------------

                        FORCEOVERLAY
                .SBTTL  SECTION NUMBER 0C
            000000      .PSECT  OVR14
```

```
4907            .SBTTL  T2D     ALU ALEG DATA PATH "RAMX_D"
                ;;*****************************************************************************
                ;++
                ;TEST   2D      ALU ALEG DATA PATH "RAMX_D"
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THE D INPUT TO THE RAMX AND THE DFMX INPUT
                ;       TO THE QMX. THIS IS DONE BY LOADING THE Q REGISTER WITH THE DATA PATTERN,
                ;       OVER THE ID BUS, TRANSFERING THE Q TO THE D (THRU THE DAL), CHANGING THE
                ;       CONTENTS OF THE Q REGISTER, THEN LOADING THE THE Q REGISTER WITH THE
                ;       D REG, THRU THE ALU, THEN TRANSFERING THE Q TO THE D REG, AND READING
                ;       AND CHECKING THE DATA. THE CONTENTS OF THE Q REGISTER MUST BE CHANGED
                ;       SO THAT IF THE RAMX SELECT FAILS, THE ERROR WILL BE DETECTED.
                ;
                ;       SUBTST 1 - FLOAT A ZERO THRU THE DATA PATTH
                ;       SUBTST 2 - FLOAT A ONE THRU THE DATA PATH
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE RAMX SELECT LOGIC, THE Q MUX SELECT LOGIC, AND
                ;       THE Q REG CLOCK CONTROL ON THE DAP MODULE, AND THE DATA PATTH FROM THE
                ;       D REG TO THE RAMX ON THE DBP, DCP, AND DDP MODULES.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED CONTENTS OF THE D REG
                ;             RECEIVED CONTENTS OF THE D REG
                ;             LOOP COUNT -- INDICATES WHAT BIT POSITION THE FLOATING ONE OR ZERO
                ;               IS IN, I.E. 1=BIT 0, 2=BIT 1, 3= BIT 2, ETC.
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC74--DATA IN D REG IS BEING GATED THRU RAMX INTO Q REG
                ;       SUBTST 2 - SYNC75--DATA IN D REG IS BEING GATED THRU RAMX INTO Q REG
                ;--
                ;;*****************************************************************************
    000034      T2D:
4911 000040             INITIALIZE
4912 000042             BLKMIC  TMPX60,,10000,2 ; LOAD INSTRUCTION TO PUT Q INTO D THRU DAL
4913 000056             BLKMIC  TMP60,,10002,2  ; LOAD TEST SEQUENCE INSTRUCTION
4914
4915 000072             SUBTEST
                ;////////////////////////////////////////////////////////////////////////////
    000072      T2DS1:
4916                    ;+
4917                    ; FIRST FLOAT A ZERO
4918                    ;-
4919
4920 000074             LOOP    I,1,32
4921 000106             ERLOOP
4922 000110             FLTZRO  TMPX61,I        ; GENERATE THE TEST PATTERN
4923 000116             FLTONE  TMP61,I         ; GENERATE COMPLIMENT TEST PATTERN
4924 000124             INITIALIZE
4925 000126             LDIDREG RWDQ,TMPX61     ; LOAD TEST PATTERN INTO Q REG
4926 000134             FETCH   10000,,NONOP    ; START THE TEST
4927 000144             CLOCK   4               ; PUT Q INTO D REG (THRU DAL)
4928 000150             LDIDREG RWDQ,TMP61      ; PUT COMPLIMENT PATTERN IN Q
4929 000156     SYNC74: CLOCK   8               ; EXECUTE THE TEST
4930 000162             READID  RWDQ            ; READ THE D REG
4931 000166             CMPCAD  EQ,IDREGLO,TMPX61 ; CHECK THE DATA
```

```
 4932 000212            IFERROR 106,DP3          ; BIT STUCK IN ALU DATA PATH
 4933 000220            ENDLOOP I                ; CONTINUE
 4934 000224            SKIP    SUBTST
 4935
 4936
 4937           ;+
 4938           ; FIND OUT WHICH BYTE IS BAD
 4939           ;-
 4940
 4941 000230  DP3:      MASK    TMPX61,OBYTE0    ; ONLY WANT BYTE 0
 4942 000236            CMPCAM  ,IDREGLO,OBYTE0,,TMPX61 ; IS BYTE 0 BAD?
 4943 000252            CHKPNT  4$               ; BRANCH IF NO
 4944 000260            FLTZRO  TMPX61,I         ; REGENERATE TEST PATTERN
 4945 000266            MASK    TMPX61,OBYTE1    ; ONLY WANT BYTE 1
 4946 000274            CMPCAM  ,IDREGLO,OBYTE1,,TMPX61 ; IS BYTE 1 BAD?
 4947 000310            CHKPNT  5$               ; BRANCH IF NO
 4948 000316            CMPCA   ,IDREGHI,TMPX61+2 ; ARE BYTES 2,3 BAD?
 4949 000330            CHKPNT  6$               ; BRANCH IF NO
 4950 000336            REPORT  <DAP>            ; ALL THE BYTES ARE BAD
 4951 000346  6$:       REPORT  <DCP,DDP>        ; BYTES 0 AND 1 BAD, 2,3 ARE GOOD
 4952 000356  5$:       REPORT  <DCP>            ; BYTE 0 BAD, BYTE 1 GOOD
 4953
 4954 000366  4$:       FLTZRO  TMPX61,I         ; REGENERATE TEST PATTERN
 4955 000374            MASK    TMPX61,OBYTE1    ; ONLY WANT BYTE 1
 4956 000402            CMPCAM  ,IDREGLO,OBYTE1,,TMPX61 ; IS BYTE 1 BAD?
 4957 000416            CHKPNT  7$               ; BRANCH IF NO
 4958 000424            REPORT  <DDP>            ; BYTE 0 GOOD, BYTE 1 BAD
 4959 000434  7$:       CMPCA   ,IDREGHI,TMPX61+2 ; BYTES 2,3 BAD?
 4960 000446            CHKPNT  8$               ; BRANCH IF NO
 4961 000454            REPORT  <DEP>            ; BYTES 0,1 GOOD, BYTES 2,3 BAD
 4962 000464  8$:       REPORT  <DAP,DBP,DCP,DDP,DEP> ; ALL BYTES GOOD (SHOULD NEVER HAPPEN)
 4963
 4964 000474            SUBTEST
              ;////////////////////////////////////////////////////////////////////////////////
      000474  T2DS2:
 4965           ;+
 4966           ; NOW FLOAT A ONE PATTERN
 4967           ;-
 4968
 4969 000476            LOOP    I,1,32
 4970 000510            ERLOOP
 4971 000512            FLTONE  TMPX61,I                 ; GENERATE THE TEST PATTERN
 4972 000520            FLTZRO  TMP61,I          ; GEN COMPLIMENT TEST PATTERN
 4973 000526            INITIALIZE
 4974 000530            LDIDREG RWDQ,TMPX61      ; PUT TEST PATTERN IN Q REG
 4975 000536            FETCH   10000,,NONOP
 4976 000546            CLOCK   4                ; TRANSFER PATTERN TO D REG
 4977 000552            LDIDREG RWDQ,TMP61       ; PUT COMPLIMENT PATTERN IN Q REG
 4978 000560  SYNC75:   CLOCK   8                ; EXECUTE THE TEST
 4979 000564            READID  RWDQ             ; GET THE TEST PATTERN BACK
 4980 000570            CMPCAD  EQ,IDREGLO,TMPX61 ; CHECK THE RECEIVED PATTERN
 4981 000614            IFERROR 107,DP3A         ; BIT STUCK IN DATA PATH
 4982 000622            ENDLOOP I                ; CONTINUE
 4983 000626            SKIP
 4984
 4985           ;+
 4986           ; FIND OUT WHICH BYTE IS BAD
```

```
4987            ;-
4988
4989 000632  DP3A:   MASK    TMPX61,OBYTE0    ; ONLY WANT BYTE 0
4990 000640          CMPCAM   ,IDREGLO,OBYTE0,,TMPX61 ; IS BYTE 0 BAD?
4991 000654          CHKPNT   4$               ; BRANCH IF NO
4992 000662          FLTONE   TMPX61,I         ; REGENERATE TEST PATTERN
4993 000670          MASK     TMPX61,OBYTE1    ; ONLY WANT BYTE 1
4994 000676          CMPCAM   ,IDREGLO,OBYTE1,,TMPX61 ; IS BYTE 1 BAD?
4995 000712          CHKPNT   5$               ; BRANCH IF NO
4996 000720          CMPCA    ,IDREGHI,TMPX61+2 ; ARE BYTES 2,3 BAD?
4997 000732          CHKPNT   6$               ; BRANCH IF NO
4998 000740          REPORT   <DAP>            ; ALL THE BYTES ARE BAD
4999 000750  6$:     REPORT   <DCP,DDP>        ; BYTES 0 AND 1 BAD, 2,3 ARE GOOD
5000 000760  5$:     REPORT   <DCP>            ; BYTE 0 BAD, BYTE 1 GOOD
5001
5002 000770  4$:     FLTONE   TMPX61,I         ; REGENERATE TEST PATTERN
5003 000776          MASK     TMPX61,OBYTE1    ; ONLY WANT BYTE 1
5004 001004          CMPCAM   ,IDREGLO,OBYTE1,,TMPX61 ; IS BYTE 1 BAD?
5005 001020          CHKPNT   7$               ; BRANCH IF NO
5006 001026          REPORT   <DDP>            ; BYTE 0 GOOD, BYTE 1 BAD
5007 001036  7$:     CMPCA    ,IDREGHI,TMPX61+2 ; BYTES 2,3 BAD?
5008 001050          CHKPNT   8$               ; BRANCH IF NO
5009 001056          REPORT   <DAP>            ; BYTES 0,1 GOOD, BYTES 2,3 BAD
5010 001066  8$:     REPORT   <DAP,DBP,DCP,DDP,DEP> ; ALL BYTES GOOD (SHOULD NEVER HAPPEN)
5011
5012 001076          TESTDAT
                ;-------------------------------------------------------------------------
5013            ;*1000: DK/Q,J/1000
5014 001102  TMPX60: .WORD    10001
5015 001104          .WORD    0
5016 001106          .WORD    4000
5017 001110          .WORD    600
5018 001112          .WORD    74
5019 001114          .WORD    6000
5020
5021            ;*1001: BEN/0,J/1002
5022 001116          .WORD    10002
5023 001120          .WORD    0
5024 001122          .WORD    4000
5025 001124          .WORD    600
5026 001126          .WORD    74
5027 001130          .WORD    0
5028
5029 001132  TMP60:
5030            ;*1002: RAMX/D,AMX/RAMX,ALU/A,SHF/ALU,QK/SHF,J/1003 ; TRANSFER D TO Q REG
5031 001132          .WORD    10003
5032 001134          .WORD    0
5033 001136          .WORD    4000
5034 001140          .WORD    700
5035 001142          .WORD    000074
5036 001144          .WORD    1
5037
5038            ;*1003: DK/Q,J/1003              ; TRANSFER Q TO D REG
5039 001146          .WORD    10003
5040 001150          .WORD    0
5041 001152          .WORD    4000
5042 001154          .WORD    600
```

T2D ALU ALEG DATA PATH 'RAMX_D'

```
 5043 001156            .WORD   000074
 5044 001160            .WORD   6000
 5045
 5046 001162  TMP61:    0,0
 5047 001166  TMPX61:   .WORD   0,0
 5048 001172  T27L1:    .WORD   SBC
 5049 001174  OBYTE0:   .WORD   377
 5050 001176  OBYTE1:   .WORD   177400
 5051 001200            ENDDAT
             ;-------------------------------------------------------------------------------
 5052
```

```
                .SBTTL  T2E     DATA PATH ALU BLEG 'RBMX_Q'
                ;*****************************************************************
                ;++
                ;TEST   2E      DATA PATH ALU BLEG 'RBMX_Q'
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THE BLEG OF THE ALU BY FLOATING A ONE AND A
                ;       ZERO THRU THE Q REG, ROUTING THE Q REG THRU THE RBMX, THRU THE
                ;       ALU, TO THE D REG, AND CHECKING THE DATA ON THE ID BUS.
                ;
                ;       SUBTST 1 - FLOAT A ZERO THRU THE B LEG
                ;       SUBTST 2 - FLOAT A ONE THRU THE B LEG
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE RBMX, THE BMX, AND THE ALU CONTROL LOGIC
                ;       ON THE DAP MODULE AND THE RBMX, AND BMX DATA PATH ON THE DBP,
                ;       DCP, AND DDP MODULES.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED CONTENTS OF THE D REGISTER
                ;             RECEIVED CONTENTS OF THE D REGISTER
                ;             LOOP COUNT -- INDICATES THE BIT POSITION OF THE FLOATING ONE OR
                ;               ZERO, I.E. 1=BIT 0, 2=BIT 1, 3=BIT 2, ETC.
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC76--DATA IS BEING GATED THRU ALU B LEG
                ;       SUBTST 2 - SYNC77--DATA IS BEING GATED THRU ALU B LEG
                ;--
                ;*****************************************************************
    001200      T2E:
5083 001204             INITIALIZE
5084 001206             BLKMIC  TMP62,,10000,1  ; LOAD THE TEST INSTRUCTION
5085
5086 001222             SUBTEST
                ;/////////////////////////////////////////////////////////////////
    001222      T2ES1:
5087            ;+
5088            ; FIRST FLOAT A ZERO THRU THE B LEG
5089            ;-
5090
5091 001224              LOOP    I,1,32
5092 001236              ERLOOP
5093 001240              FLTZRO  T28L1,I         ; GENERATE THE TEST PATTERN
5094 001246              INITIALIZE
5095 001250              LDIDREG RWDQ,T28L1      ; LOAD THE TEST PATTERN
5096 001256              FETCH   10000,,NONOP
5097 001266  SYNC76:     CLOCK   4               ; EXECUTE THE TEST INSTRUCTION
5098 001272              READID  RWDQ            ; READ THE DATA BACK
5099 001276              CMPCAD 'EQ,IDREGLO,T28L1 ; CHECK THE DATA
5100 001322              IFERROR 110,DP4         ; BIT STUCK IN BLEG DATA PATH
5101 001330              ENDLOOP I               ; CONTINUE
5102 001334              SKIP    SUBTST
5103
5104
5105
5106            ;+
5107            ; FIND OUT WHICH BYTE IS BAD
```

```
                        ;-
                        ;
001340  DP4:    MASK    T28L1,PBYTE0    ; ONLY WANT BYTE 0
001346          CMPCAM  ,IDREGLO,PBYTE0,,T28L1 ; IS BYTE 0 BAD?
001362          CHKPNT  4$              ; BRANCH IF NO
001370          FLTZRO  T28L1,I         ; REGENERATE TEST PATTERN
001376          MASK    T28L1,PBYTE1    ; ONLY WANT BYTE 1
001404          CMPCAM  ,IDREGLO,PBYTE1,,T28L1 ; IS BYTE 1 BAD?
001420          CHKPNT  5$              ; BRANCH IF NO
001426          CMPCA   ,IDREGHI,T28L1+2 ; ARE BYTES 2,3 BAD?
001440          CHKPNT  6$              ; BRANCH IF NO
001446          REPORT  <DAP>           ; ALL THE BYTES ARE BAD
001456  6$:     REPORT  <DAP,DCP,DDP>   ; BYTES 0 AND 1 BAD, 2,3 ARE GOOD
001466  5$:     REPORT  <DCP>           ; BYTE 0 BAD, BYTE 1 GOOD
                        ;
001476  4$:     FLTZRO  T28L1,I         ; REGENERATE TEST PATTERN
001504          MASK    T28L1,PBYTE1    ; ONLY WANT BYTE 1
001512          CMPCAM  ,IDREGLO,PBYTE1,,T28L1 ; IS BYTE 1 BAD?
001526          CHKPNT  7$              ; BRANCH IF NO
001534          REPORT  <DDP>           ; BYTE 0 GOOD, BYTE 1 BAD
001544  7$:     CMPCA   ,IDREGHI,T28L1+2 ; BYTES 2,3 BAD?
001556          CHKPNT  8$              ; BRANCH IF NO
001564          REPORT  <DEP>           ; BYTES 0,1 GOOD, BYTES 2,3 BAD
001574  8$:     REPORT  <DAP,DBP,DCP,DDP,DEP> ; ALL BYTES GOOD (SHOULD NEVER HAPPEN)
                        ;
                        ;
001604          SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
001604  T2ES2:
                ;+
                ; NOW FLOAT A ONE THRU THE B LEG.
                ;-
                ;
001606          LOOP    I,1,32
001620          ERLOOP
001622          FLTONE  T28L1,I         ; GENERATE THE TEST PATTERN
001630          INITIALIZE
001632          LDIDREG RWDQ,T28L1      ; LOAD THE TEST PATTERN
001640          FETCH   10000,,NONOP
001650  SYNC77: CLOCK   4               ; EXECUTE THE TEST MICRO INSTRUCTION
001654          READID  RWDQ            ; READ THE D REGISTER
001660          CMPCAD  EQ,IDREGLO,T28L1 ; CHECK THE RECEIVED DATA
001704          IFERROR 111,DP4A        ; BIT STUCK IN BLEG DATA PATH
001712          ENDLOOP I               ; CONTINUE
001716          SKIP
                ;
                ;+
                ; FIND OUT WHICH BYTE IS BAD
                ;-
                ;
001722  DP4A:   MASK    T28L1,PBYTE0    ; ONLY WANT BYTE 0
001730          CMPCAM  ,IDREGLO,PBYTE0,,T28L1 ; IS BYTE 0 BAD?
001744          CHKPNT  4$              ; BRANCH IF NO
001752          FLTONE  T28L1,I         ; REGENERATE TEST PATTERN
001760          MASK    T28L1,PBYTE1    ; ONLY WANT BYTE 1
001766          CMPCAM  ,IDREGLO,PBYTE1,,T28L1 ; IS BYTE 1 BAD?
002002          CHKPNT  5$              ; BRANCH IF NO
```

```
5163 002010           CMPCA   ,IDREGHI,T28L1+2 ; ARE BYTES 2,3 BAD?
5164 002022           CHKPNT  6$              ; BRANCH IF NO
5165 002030           REPORT  <DAP>           ; ALL THE BYTES ARE BAD
5166 002040  6$:      REPORT  <DAP,DCP,DDP>   ; BYTES 0 AND 1 BAD, 2,3 ARE GOOD
5167 002050  5$:      REPORT  <DCP>           ; BYTE 0 BAD, BYTE 1 GOOD
5168
5169 002060  4$:      FLTONE  T28L1,I         ; REGENERATE TEST PATTERN
5170 002066           MASK    T28L1,PBYTE1    ; ONLY WANT BYTE 1
5171 002074           CMPCAM  ,IDREGLO,PBYTE1,,T28L1 ; IS BYTE 1 BAD?
5172 002110           CHKPNT  7$              ; BRANCH IF NO
5173 002116           REPORT  <DDP>           ; BYTE 0 GOOD, BYTE 1 BAD
5174 002126  7$:      CMPCA   ,IDREGHI,T28L1+2 ; BYTES 2,3 BAD?
5175 002140           CHKPNT  8$              ; BRANCH IF NO
5176 002146           REPORT  <DEP>           ; BYTES 0,1 GOOD, BYTES 2,3 BAD
5177 002156  8$:      REPORT  <DAP,DBP,DCP,DDP,DEP> ; ALL BYTES GOOD (SHOULD NEVER HAPPEN)
5178
5179 002166           TESTDAT
                ;-------------------------------------------------------------------------------
5180 002172  TMP62:
5181                ;*1000: RBMX/Q,BMX/RBMX,ALU/B,SHF/ALU,DK/SHF,J/1000 ; Q TO D THRU B MUX
5182 002172           .WORD   10000
5183 002174           .WORD   0
5184 002176           .WORD   4000
5185 002200           .WORD   600
5186 002202           .WORD   000070
5187 002204           .WORD   4034
5188 002206  T28L1:   .WORD   0,0
5189 002212  T28L2:   .WORD   SBC
5190 002214  PBYTE0:  .WORD   377
5191 002216  PBYTE1:  .WORD   177400
5192 002220           ENDDAT
                ;-------------------------------------------------------------------------------
5193
5194 002220           FORCEOVERLAY
                .SBTTL  SECTION NUMBER 0D
     000000           .PSECT  OVR15
5195
```

```
5226            .SBTTL  T2F     DATA PATH ALU BLEG ''RBMX_D''
                ;;*****************************************************************************
                ;++
                ;TEST   2F      DATA PATH ALU BLEG ''RBMX_D''
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THE D SIDE OF THE RBMX. THIS IS DONE BY FLOATING A ZERO
                ;       AND A ONE THRU THE Q REG, TRANSFERING THE Q TO THE D REG, WRITING THE
                ;       COMPLIMENT TEST PATTERN INTO THE Q REG, TRANSFERING THE D REG TO THE
                ;       Q REG THRU THE ALU B LEG, THEN TRANSFERING THE Q TO THE D (THRU THE
                ;       DAL) AND CHECKING THE DATA.
                ;
                ;       THE COMPLIMENT PATTERN IS WRITTEN IN THE Q REG SO IF THE RBMX SELECT
                ;       FAILS, THE ERROR WILL BE DETECTED.
                ;
                ;       SUBTST 1 - FLOAT A ZERO PATTERN
                ;       SUBTST 2 - FLOAT A ONE PATTERN
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE RBMX SELECT LOGIC ON THE DAP MODULE, AND THE
                ;       D INPUT LEG TO THE RBMX ON THE DBP, DCP, AND DDP MODULES.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED CONTENTS OF THE D REGISTER
                ;             RECEIVED CONTENTS OF THE D REGISTER
                ;             LOOP COUNT -- INDICATES WHAT BIT POSITION THE FLOATING PATTERN
                ;               IS IN, I.E. 1=BIT 0, 2=BIT 1, 3=BIT 2, ETC.
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC78--TEST PATTERN IS BEING GATED THRU RBMX
                ;       SUBTST 2 - SYNC79--TEST PATTERN IS BEING GATED THRU RBMX
                ;--
                ;;*****************************************************************************
    000034      T2F:
5230 000040             INITIALIZE
5231 000042             BLKMIC  TMP63,,10000,4  ; LOAD THE INSTRUCTION SEQUENCE
5232
5233 000056             SUBTEST
                ;//////////////////////////////////////////////////////////////////////////////
    000056      T2FS1:
5234            ;+
5235            ; FIRST FLOAT A ZERO THRU THE RBMX
5236            ;-
5237
5238 000060             LOOP    I,1,32
5239 000072             ERLOOP
5240 000074             FLTZRO  TMP65,I         ; GENERATE THE TEST PATTERN
5241 000102             FLTONE  TMP66,I         ; GEN COMPLIMENT TEST PATTERN
5242 000110             INITIALIZE
5243 000112             LDIDREG RWDQ,TMP65      ; LOAD THE TEST PATTERN
5244 000120             FETCH   10000,,NONOP
5245 000130             CLOCK   4               ; PUT Q INTO D
5246 000134             LDIDREG RWDQ,TMP66      ; PUT COMPLIMENT PATTERN IN Q AND EXECUTE TEST INSTR
5247 000142     SYNC78: CLOCK   8               ; PUT Q INTO D
5248 000146             READID  RWDQ            ; READ THE DATA BACK
5249 000152             CMPCAD  EQ,IDREGLO,TMP65 ; CHECK THE DATA
5250 000176             IFERROR 112,DP5         ; BIT STUCK IN RBMX, D INPUT
```

```
5251 000204            ENDLOOP I                    ; CONTINUE
5252 000210            SKIP    SUBTST
5253
5254
5255
5256            ;+
5257            ; FIND OUT WHICH BYTE IS BAD
5258            ;-
5259
5260 000214  DP5:     MASK    TMP65,QBYTE0         ; ONLY WANT BYTE 0
5261 000222           CMPCAM  ,IDREGLO,QBYTE0,,TMP65 ; IS BYTE 0 BAD?
5262 000236           CHKPNT  4$                   ; BRANCH IF NO
5263 000244           FLTZRO  TMP65,I              ; REGENERATE TEST PATTERN
5264 000252           MASK    TMP65,QBYTE1         ; ONLY WANT BYTE 1
5265 000260           CMPCAM  ,IDREGLO,QBYTE1,,TMP65 ; IS BYTE 1 BAD?
5266 000274           CHKPNT  5$                   ; BRANCH IF NO
5267 000302           CMPCA   ,IDREGHI,TMP65+2     ; ARE BYTES 2,3 BAD?
5268 000314           CHKPNT  6$                   ; BRANCH IF NO
5269 000322           REPORT  <DAP>                ; ALL THE BYTES ARE BAD
5270 000332  6$:      REPORT  <DAP,DCP,DDP>        ; BYTES 0 AND 1 BAD, 2,3 ARE GOOD
5271 000342  5$:      REPORT  <DCP>                ; BYTE 0 BAD, BYTE 1 GOOD
5272
5273 000352  4$:      FLTZRO  TMP65,I              ; REGENERATE TEST PATTERN
5274 000360           MASK    TMP65,QBYTE1         ; ONLY WANT BYTE 1
5275 000366           CMPCAM  ,IDREGLO,QBYTE1,,TMP65 ; IS BYTE 1 BAD?
5276 000402           CHKPNT  7$                   ; BRANCH IF NO
5277 000410           REPORT  <DDP>                ; BYTE 0 GOOD, BYTE 1 BAD
5278 000420  7$:      CMPCA   ,IDREGHI,TMP65+2     ; BYTES 2,3 BAD?
5279 000432           CHKPNT  8$                   ; BRANCH IF NO
5280 000440           REPORT  <DEP>                ; BYTES 0,1 GOOD, BYTES 2,3 BAD
5281 000450  8$:      REPORT  <DAP,DBP,DCP,DDP,DEP> ; ALL BYTES GOOD (SHOULD NEVER HAPPEN)
5282
5283
5284 000460            SUBTEST
            ;////////////////////////////////////////////////////////////////////////////////
     000460 T2FS2:
5285            ;+
5286            ; NOW FLOAT A ONE THRU THE RBMX D INPUT
5287            ;-
5288
5289 000462            LOOP    I,1,32
5290 000474            ERLOOP
5291 000476            FLTONE  TMP65,I              ; GENERATE THE TEST PATTERN
5292 000504            FLTZRO  TMP66,I              ; GEN THE COMPLIMENT PATTERN
5293 000512            INITIALIZE
5294 000514            LDIDREG RWDQ,TMP65           ; LOAD THE TEST PATTERN
5295 000522            FETCH   10000,,NONOP
5296 000532            CLOCK   4                    ; PUT PATTERN INTO D REG
5297 000536            LDIDREG RWDQ,TMP66           ; PUT COMPLIMENT PATTERN INTO Q
5298 000544  SYNC79:   CLOCK   8                    ; EXECUTE THE TEST
5299 000550            READID  RWDQ
5300 000554            CMPCAD  EQ,IDREGLO,TMP65 ; CHECK THE RECEIVED DATA
5301 000600            IFERROR 113,DP5A             ; BIT STUCK IN RBMX D INPUT
5302 000606            ENDLOOP I                    ; CONTINUE
5303 000612            SKIP
5304
5305            ;+
```

```
                        ; FIND OUT WHICH BYTE IS BAD
                        ;-
                        
000616  DP5A:   MASK    TMP65,QBYTE0    ; ONLY WANT BYTE 0
000624          CMPCAM  ,IDREGLO,QBYTE0,,TMP65 ; IS BYTE 0 BAD?
000640          CHKPNT  4$              ; BRANCH IF NO
000646          FLTONE  TMP65,I         ; REGENERATE TEST PATTERN
000654          MASK    TMP65,QBYTE1    ; ONLY WANT BYTE 1
000662          CMPCAM  ,IDREGLO,QBYTE1,,TMP65 ; IS BYTE 1 BAD?
000676          CHKPNT  5$              ; BRANCH IF NO
000704          CMPCA   ,IDREGHI,TMP65+2 ; ARE BYTES 2,3 BAD?
000716          CHKPNT  6$              ; BRANCH IF NO
000724          REPORT  <DAP>           ; ALL THE BYTES ARE BAD
000734  6$:     REPORT  <DAP,DCP,DDP>   ; BYTES 0 AND 1 BAD, 2,3 ARE GOOD
000744  5$:     REPORT  <DCP>           ; BYTE 0 BAD, BYTE 1 GOOD
                
000754  4$:     FLTONE  TMP65,I         ; REGENERATE TEST PATTERN
000762          MASK    TMP65,QBYTE1    ; ONLY WANT BYTE 1
000770          CMPCAM  ,IDREGLO,QBYTE1,,TMP65 ; IS BYTE 1 BAD?
001004          CHKPNT  7$              ; BRANCH IF NO
001012          REPORT  <DDP>           ; BYTE 0 GOOD, BYTE 1 BAD
001022  7$:     CMPCA   ,IDREGHI,TMP65+2 ; BYTES 2,3 BAD?
001034          CHKPNT  8$              ; BRANCH IF NO
001042          REPORT  <DEP>           ; BYTES 0,1 GOOD, BYTES 2,3 BAD
001052  8$:     REPORT  <DAP,DBP,DCP,DDP,DEP> ; ALL BYTES GOOD (SHOULD NEVER HAPPEN)
                
001062          TESTDAT
                ;-------------------------------------------------------------------------
001066  TMP63:
                ;*1000: DK/Q,J/1001              ; LOAD D FROM Q THRU THE DAL
001066          .WORD   10001
001070          .WORD   0
001072          .WORD   4000
001074          .WORD   600
001076          .WORD   000074
001100          .WORD   6000
                
                ;*1001: BEN/0,J/1002             ; NOP
001102          .WORD   10002
001104          .WORD   0
001106          .WORD   4000
001110          .WORD   600
001112          .WORD   000074
001114          .WORD   0
                
                ;*1002: RBMX/D,BMX/RBMX,ALU/B,SHF/ALU,QK/SHF,J/1003 ; Q GETS D THRU RBMX
001116          .WORD   10003
001120          .WORD   0
001122          .WORD   4000
001124          .WORD   700
001126          .WORD   20070
001130          .WORD   34
                
                ;*1003: DK/Q,J/1003              ; LOAD D FROM Q THRU THE DAL
001132          .WORD   10003
001134          .WORD   0
001136          .WORD   4000
```

T2F DATA PATH ALU BLEG ''RBMX_D''

```
 5362 001140            .WORD   600
 5363 001142            .WORD   000074
 5364 001144            .WORD   6000
 5365
 5366 001146  TMP65:    .WORD   0,0             ; WORKING LOCATION
 5367 001152  TMP66:    .WORD   0,0             ; WORKING LOCATION
 5368 001156  TMP67:    .WORD   SBC
 5369 001160  QBYTE0:   .WORD   377
 5370 001162  QBYTE1:   .WORD   177400
 5371 001164            ENDDAT
              ;-------------------------------------------------------------------------------
 5372
 5373 001164            FORCEOVERLAY
              .SBTTL    SECTION NUMBER 0E
      000000            .PSECT  OVR16
 5374
```

```
                        .SBTTL  T30     DATA PATH ALU ARITHMETIC MODE
                        ;;******************************************************************************
                        ;++
                        ;TEST   30      DATA PATH ALU ARITHMETIC MODE
                        ;
                        ; TEST DESCRIPTION
                        ;       THIS TEST CHECKS THE A+B FUNCTION OF THE ALU. THE
                        ;       A+B FUNCTION IS TESTED WITH 32 DATA PATTERNS TO CHECK THE LOOK-
                        ;       AHEAD CARRY LOGIC ON THE ALU.
                        ;
                        ;       THIS IS DONE BY LOADING THE BLEG DATA INTO THE D REGISTER AND THE
                        ;       ALEG DATA INTO THE Q REGISTER, EXECUTING THE ALU FUNCTION AND PUTTING
                        ;       THE RESULT IN THE D REGISTER WHERE IT IS READ AND CHECKED.
                        ;
                        ;       SUBTST 1 - CHECK 32 DATA PATTERNS IN A+B MODE.
                        ;       SUBTST 2 - CHECK 16 DATA PATTERNS IN A+B MODE
                        ;
                        ; LOGIC DESCRIPTION
                        ;       THIS TEST CHECKS THE ALU FUNCTION CONTROL LOGIC ON THE DAP MODULE AND
                        ;       THE LOOK AHEAD CARRY LOGIC FOR THE ALU ON THE DCP, DDP, AND DEP MODULES.
                        ;
                        ; ERROR DESCRIPTION
                        ;
                        ;       IF THE OUTPUT DATA IS 4 DIGITS IN LENGTH, IT REPRESENTS THE V BUS
                        ;       SIGNAL FOR "ALU CARRY 31 L". IF IT IS 8 DIGITS IN LENGTH IT REPRESENTS
                        ;       THE ALU OUTPUT.
                        ;
                        ;       DATA: EXPECTED
                        ;             RECEIVED
                        ;             LOOP COUNT -- INDICATES WHICH DATA PATTERN IS BEING USED.
                        ;
                        ; SYNC POINT DESCRIPTION
                        ;       SUBTST 1 - SYNC7A--DATA IS BEING GATED THRU THE ALU
                        ;       SUBTST 2 - SYNC7B--DATA IS BEING GATED THRU THE ALU
                        ;--
                        ;;******************************************************************************
            000034      T30:
5412 000040                     INITIALIZE
5413 000042                     BLKMIC  TMP70,,10000,3  ; LOAD THE MICRO WORDS TO DO AN A+B
5414
5415 000056                     SUBTEST
                        ;////////////////////////////////////////////////////////////////////////////////
            000056      T30S1:
5416                    ;+
5417                    ; CHECK 32 PATTERNS FOR BYTES 2 AND 3
5418                    ;-
5419
5420 000060                     LOOP    I,1,32          ; CHECK THE FIRST 32 PATTERNS
5421 000072                     ERLOOP
5422 000074                     INITIALIZE
5423 000076                     LDIDREG RWDQ,TMP71,I    ; LOAD THE BLEG DATA
5424 000104                     FETCH   10000,,NONOP
5425 000114                     CLOCK   4               ; PUT IT IN THE D REGISTER
5426 000120                     LDIDREG RWDQ,TMP72,I    ; LOAD THE ALEG TEST DATA
5427 000126     SYNC7A:         CLOCK   2               ; EXECUTE THE A+B
5428 000132                     TSTVB   T2FV3           ; CHECK "ALU CARRY 31" ACTIVE
5429 000140                     IFERROR ,DP6A
```

```
5430 000146            CLOCK   2                ; FINISH THE ALU FUNCTION
5431 000152            READID  RWDQ             ; READ THE RESULT
5432 000156            CMPCAD  EQ,IDREGLO,TMP73 ; CHECK THE RESULT
5433 000202            IFERROR 114,DP6          ; ALU A+B FAILED
5434 000210            ENDLOOP I                ; CONTINUE
5435 000214            SKIP    SUBTST
5436
5437
5438 000220  DP6A:     REPORT  <DEP,DCP>        ; ALU CARRY 31 DID NOT GO ACTIVE
5439
5440
5441 000230  DP6:      CMPCAM  ,IDREGLO,RBYTE0,,TMP73 ; IS BYTE 0 OK?
5442 000244            CHKPNT  1$               ; BRANCH IF YES
5443 000252            CMPCAM  ,IDREGLO,RBYTE1,,TMP73 ; BYTE 1 OK?
5444 000266            CHKPNT  2$               ; BRANCH IF YES
5445 000274            CMPCA   ,IDREGHI,TMP73+2 ; BYTE 2,3 OK?
5446 000306            CHKPNT  3$               ; BRANCH IF YES
5447 000314            REPORT  <DAP>            ; ALL BYTES ARE BAD (CHECK ALU FUNCTION)
5448 000324  3$:       REPORT  <DAP,DCP>        ; BYTES 0 AND 1 BAD, 2,3 ARE GOOD
5449 000334  2$:       CMPCAM  ,IDREGLO,RBYTE0,,T2FL6 ; IS BIT 0 BAD?
5450 000350            CHKPNT  8$               ; BRANCH IF YES
5451 000356            REPORT  <DCP>            ; BYTE 0 BAD, BYTE 1 GOOD
5452 000366  8$:       REPORT  <DAP,DCP>        ; BYTE 0 BIT 0 BAD, BYTE 1 GOOD
5453
5454           ;+
5455           ; BYTE 0 IS OK SO LETS CHECK BYTES 1 AND 2,3
5456           ;-
5457
5458 000376  1$:       CMPCAM  ,IDREGLO,RBYTE1,,TMP73 ; IS BYTE 1 OK?
5459 000412            CHKPNT  4$               ; BRANCH IF YES
5460
5461           ;+
5462           ; BYTE 0 IS GOOD BUT BYTE 1 IS BAD. REEXECUTE THE FUNCTION AND CHECK
5463           ; THAT ALU CARRY 7 IS INACTIVE.
5464           ;-
5465
5466 000420            INITIALIZE
5467 000422            LDIDREG RWDQ,TMP71,I     ; LOAD THE B LEG DATA
5468 000430            FETCH   10000,,NONOP     ; START THE SEQUENCER
5469 000440            CLOCK   4                ; PUT DATA IN D REG
5470 000444            LDIDREG RWDQ,TMP72,I     ; LOAD THE A LEG DATA
5471 000452            CLOCK   2                ; STOP IN MIDDLE OF A+B
5472 000456            TSTVB   T2FV1            ; CHECK "ALU CARRY 07" INACTIVE
5473 000464            CHKPNT  5$               ; BRANCH IF OK
5474 000472            REPORT  <DCP>            ; "ALU CARRY 07" NOT INACTIVE
5475 000502  5$:       REPORT  <DDP>            ; CARRY INACTIVE BUT BYTE 1 WAS BAD
5476
5477           ;+
5478           ; BYTES 0 AND 1 WERE OK, CHECK BYTES 2,3
5479           ;-
5480
5481 000512  4$:       CMPCA   ,IDREGHI,TMP73+2 ; BYTES 2,3 OK?
5482 000524            CHKPNT  6$               ; BRANCH IF YES
5483
5484           ;+
5485           ; BYTES 0 AND 1 GOOD BUT BYTES 2,3 BAD. CHECK THAT "ALU CARRY 15" IS
5486           ; INACTIVE.
```

```
                        ;-
                        ;
                                INITIALIZE
                                LDIDREG RWDQ,TMP71,I    ; LOAD THE BLEG DATA
                                FETCH   10000,,NONOP    ; START THE SEQUENCER
                                CLOCK   4               ; PUT DATA IN D REG
                                LDIDREG RWDQ,TMP72,I    ; LOAD THE A LEG DATA
                                CLOCK   2               ; STOP IN MIDDLE OF A+B
                                TSTVB   T2FV2           ; CHECK IF "ALU CARRY 15" IS INACTIVE
                                CHKPNT  7$              ; BRANCH IF IT IS
                                REPORT  <DCP,DDP>       ; "ALU CARRY 15" IS NOT INACTIVE
                        7$:     REPORT  <DEP>           ; "ALU CARRY 15" INACTIVE BUT RESULT WRONG
                        6$:     REPORT  <DAP,DBP,DCP,DDP,DEP> ; ALL BYTES GOOD (SHOULD NEVER HAPPEN)
                        ;
                                SUBTEST
                        ;////////////////////////////////////////////////////////////////////////////////
                        T30S2:
                        ;+
                        ; CHECK 16 PATTERNS FOR BYTE 1
                        ;-
                        ;
                                LOOP    I,1,16
                                ERLOOP
                                INITIALIZE
                                LDIDREG RWDQ,TMP74,I    ; LOAD THE BLEG TEST DATA
                                FETCH   10000,,NONOP
                                CLOCK   4               ; PUT IT IN THE D REG
                                LDIDREG RWDQ,TMP75,I    ; LOAD THE ALEG TEST DATA
                        SYNC7B: CLOCK   2               ; EXECUTE THE A+B
                                TSTVB   T2FV3           ; CHECK THAT "ALU CARRY 31" IS ACTIVE
                                IFERROR ,DP6B           ; CARRY LOGIC FAILED
                                CLOCK   2               ; COMPLETE THE ALU FUNCTION
                                READID  RWDQ
                                CMPCAD  EQ,IDREGLO,TMP73 ; CHECK THE RESULT
                                IFERROR 115,DP6C        ; ALU FAILED
                                ENDLOOP I               ; CONTINUE
                                SKIP
                        ;
                        DP6B:   TSTVB   T2FV4           ; CHECK "ALU CARRY 15" ACTIVE
                                CHKPNT  1$              ; BRANCH IF IT IS
                                REPORT  <DDP,DCP>       ; "ALU CARRY 15" NOT ACTIVE
                        1$:     REPORT  <DEP>           ; CARRY 15 ACTIVE BUT CARRY 31 NOT ACTIVE
                        ;
                        ;
                        DP6C:   CMPCAM  ,IDREGLO,RBYTE0,,TMP73 ; BYTE 0 OK?
                                CHKPNT  1$              ; BRANCH IF YES
                                CMPCAM  ,IDREGLO,RBYTE1,,TMP73 ; BYTE 1 OK?
                                CHKPNT  2$              ; BRANCH IF YES
                                CMPCA   ,IDREGHI,TMP73+2 ; BYTE 2,3 OK?
                                CHKPNT  3$              ; BRANCH IF YES
                                REPORT  <DAP>           ; ALL BYTES BAD
                        2$:     REPORT  <DCP,DAP>       ; BYTE 0 BAD, BYTE 1 OK
                        3$:     REPORT  <DCP,DDP>       ; BYTE 0,1 BAD, BYTES 2,3 OK
                        ;
                        1$:     CMPCAM  ,IDREGLO,RBYTE1,,TMP73 ; BYTE 1 OK?
                                CHKPNT  4$              ; BRANCH IF YES
                                REPORT  <DDP,DCP>       ; BYTE 0 GOOD, BYTE 1 BAD
```

```
5542 001200  4$:     CMPCA   ,IDREGHI,TMP73+2 ; BYTES 2,3 OK?
5543 001212          CHKPNT  5$               ; BRANCH IF YES
5544 001220          REPORT  <DCP,DEP>        ; BYTES 0,1 GOOD, BYTES 2,3 BAD
5545 001230  5$:     REPORT  <DAP,DCP,DDP,DEP> ; ALL BYTES GOOD
5546
5547
5548
5549 001240          TESTDAT
                    ;-------------------------------------------------------------------------------
5550 001244  TMP70:
5551                 ;*1000: DK/Q,J/1001              ; LOAD D WITH Q THRU DAL
5552 001244                  .WORD   10001
5553 001246                  .WORD   0
5554 001250                  .WORD   4000
5555 001252                  .WORD   600
5556 001254                  .WORD   000074
5557 001256                  .WORD   6000
5558
5559                 ;*1001: BEN/0,J/1002             ; NOP
5560 001260                  .WORD   10002
5561 001262                  .WORD   0
5562 001264                  .WORD   4000
5563 001266                  .WORD   600
5564 001270                  .WORD   000074
5565 001272                  .WORD   0
5566
5567                 ;*1002: RAMX/Q,RBMX/D,AMX/RAMX,BMX/RBMX,ALU/A+B,SHF/ALU,DK/SHF,J/1002
5568 001274                  .WORD   10002
5569 001276                  .WORD   0
5570 001300                  .WORD   4000
5571 001302                  .WORD   600
5572 001304                  .WORD   20024
5573 001306                  .WORD   4035
5574
5575                 ;+
5576                 ; FOLLOWING IS THE B LEG DATA FOR SUBTEST 1
5577                 ;-
5578
5579 001310  TMP71:  .WORD   0,100000         ; HEX 0,8000
5580 001314          .WORD   0,40000          ; HEX 0,4000
5581 001320          .WORD   0,120000         ; HEX 0,A000
5582 001324          .WORD   0,50000          ; HEX 0,5000
5583 001330          .WORD   0,124000         ; HEX 0,A800
5584 001334          .WORD   0,52000          ; HEX 0,5400
5585 001340          .WORD   0,125000         ; HEX 0,AA00
5586 001344          .WORD   0,52400          ; HEX 0,5500
5587 001350          .WORD   0,125200         ; HEX 0,AA80
5588 001354          .WORD   0,52500          ; HEX 0,5540
5589 001360          .WORD   0,125240         ; HEX 0,AAA0
5590 001364          .WORD   0,52520          ; HEX 0,5550
5591 001370          .WORD   0,125250         ; HEX 0,AAA8
5592 001374          .WORD   0,52524          ; HEX 0,5554
5593 001400          .WORD   0,125252         ; HEX 0,AAAA
5594 001404          .WORD   0,52525          ; HEX 0,5555
5595 001410          .WORD   0,100000         ; HEX 0,8000
5596 001414          .WORD   0,140000         ; HEX 0,C000
5597 001420          .WORD   0,60000          ; HEX 0,6000
```

```
5598 001424            .WORD   0,130000        ; HEX 0,B000
5599 001430            .WORD   0,54000         ; HEX 0,5800
5600 001434            .WORD   0,126000        ; HEX 0,AC00
5601 001440            .WORD   0,53000         ; HEX 0,5600
5602 001444            .WORD   0,125400        ; HEX 0,AB00
5603 001450            .WORD   0,52600         ; HEX 0,5580
5604 001454            .WORD   0,125300        ; HEX 0,AAC0
5605 001460            .WORD   0,52540         ; HEX 0,5560
5606 001464            .WORD   0,125260        ; HEX 0,AAB0
5607 001470            .WORD   0,52530         ; HEX 0,5558
5608 001474            .WORD   0,125254        ; HEX 0,AAAC
5609 001500            .WORD   0,52526         ; HEX 0,5556
5610 001504            .WORD   0,125253        ; HEX 0,AAAB
5611
5612                ;+
5613                ; FOLLOWING IS THE ALEG DATA FOR SUBTEST 1
5614                ;-
5615
5616 001510    TMP72:  .WORD   0,100000        ; HEX 0,8000
5617 001514            .WORD   0,140000        ; HEX 0,C000
5618 001520            .WORD   0,60000         ; HEX 0,6000
5619 001524            .WORD   0,130000        ; HEX 0,B000
5620 001530            .WORD   0,54000         ; HEX 0,5800
5621 001534            .WORD   0,126000        ; HEX 0,AC00
5622 001540            .WORD   0,53000         ; HEX 0,5600
5623 001544            .WORD   0,125400        ; HEX 0,AB00
5624 001550            .WORD   0,52600         ; HEX 0,5580
5625 001554            .WORD   0,125300        ; HEX 0,AAC0
5626 001560            .WORD   0,52540         ; HEX 0,5560
5627 001564            .WORD   0,125260        ; HEX 0,AAB0
5628 001570            .WORD   0,52530         ; HEX 0,5558
5629 001574            .WORD   0,125254        ; HEX 0,AAAC
5630 001600            .WORD   0,52526         ; HEX 0,5556
5631 001604            .WORD   0,125253        ; HEX 0,AAAB
5632 001610            .WORD   0,100000        ; HEX 0,8000
5633 001614            .WORD   0,40000         ; HEX 0,4000
5634 001620            .WORD   0,120000        ; HEX 0,A000
5635 001624            .WORD   0,50000         ; HEX 0,5000
5636 001630            .WORD   0,124000        ; HEX 0,A800
5637 001634            .WORD   0,52000         ; HEX 0,5400
5638 001640            .WORD   0,125000        ; HEX 0,AA00
5639 001644            .WORD   0,52400         ; HEX 0,5500
5640 001650            .WORD   0,125200        ; HEX 0,AA80
5641 001654            .WORD   0,52500         ; HEX 0,5540
5642 001660            .WORD   0,125240        ; HEX 0,AAA0
5643 001664            .WORD   0,52520         ; HEX 0,5550
5644 001670            .WORD   0,125250        ; HEX 0,AAA8
5645 001674            .WORD   0,52524         ; HEX 0,5554
5646 001700            .WORD   0,125252        ; HEX 0,AAAA
5647 001704            .WORD   0,52525         ; HEX 0,5555
5648
5649                ;+
5650                ; FOLLOWING IS THE EXPECTED ALU OUTPUT FOR SUBTEST 1
5651                ;-
5652
5653 001710    TMP73:  .WORD   0,0             ; EXPECTED DATA
5654
```

```
5655                    ;+
5656                    ; FOLLOWING IS THE B LEG DATA FOR SUBTEST 2
5657                    ;-
5658
5659 001714  TMP74:    .WORD   100000,125252   ; HEX 8000,AAAA
5660 001720             .WORD   40000,52525     ; HEX 4000,5555
5661 001724             .WORD   120000,125252   ; HEX A000,AAAA
5662 001730             .WORD   50000,52525     ; HEX 5000,5555
5663 001734             .WORD   124000,125252   ; HEX A800,AAAA
5664 001740             .WORD   52000,52525     ; HEX 5400,5555
5665 001744             .WORD   125000,125252   ; HEX AA00,AAAA
5666 001750             .WORD   52400,52525     ; HEX 5500,5555
5667 001754             .WORD   100000,52525    ; HEX 8000,5555
5668 001760             .WORD   140000,125252   ; HEX C000,AAAA
5669 001764             .WORD   60000,52525     ; HEX 6000,5555
5670 001770             .WORD   130000,125252   ; HEX B000,AAAA
5671 001774             .WORD   54000,52525     ; HEX 5800,5555
5672 002000             .WORD   126000,125252   ; HEX AC00,AAAA
5673 002004             .WORD   53000,52525     ; HEX 5600,5555
5674 002010             .WORD   125400,125252   ; HEX AB00,AAAA
5675
5676                    ;+
5677                    ; FOLLOWING IS THE A LEG DATA FOR SUBTEST 2
5678                    ;-
5679
5680 002014  TMP75:    .WORD   100000,52525    ; HEX 8000,5555
5681 002020             .WORD   140000,125252   ; HEX C000,AAAA
5682 002024             .WORD   60000,52525     ; HEX 6000,5555
5683 002030             .WORD   130000,125252   ; HEX B000,AAAA
5684 002034             .WORD   54000,52525     ; HEX 5800,5555
5685 002040             .WORD   126000,125252   ; HEX AC00,AAAA
5686 002044             .WORD   53000,52525     ; HEX 5600,5555
5687 002050             .WORD   125400,125252   ; HEX AB00,AAAA
5688 002054             .WORD   100000,125252   ; HEX 8000,AAAA
5689 002060             .WORD   40000,52525     ; HEX 4000,5555
5690 002064             .WORD   120000,125252   ; HEX A000,AAAA
5691 002070             .WORD   50000,52525     ; HEX 5000,5555
5692 002074             .WORD   124000,125252   ; HEX A800,AAAA
5693 002100             .WORD   52000,52525     ; HEX 5400,5555
5694 002104             .WORD   125000,125252   ; HEX AA00,AAAA
5695 002110             .WORD   52400,52525     ; HEX 5500,5555
5696
5697 002114  T2FL1:     .WORD   SBC
5698 002116  T2FL6:     .WORD   1
5699 002120  T2FV1:     .WORD   1
5700 002122             VBUSG   1,23,1          ; ALU CARRY 07 L
5701 002124  T2FV2:     .WORD   1
5702 002126             VBUSG   1,22,1          ; ALU CARRY 15 L
5703 002130  T2FV3:     .WORD   1
5704 002132             VBUSG   1,21,0          ; ALU CARRY 31 L
5705 002134  T2FV4:     .WORD   1
5706 002136             VBUSG   1,22,0          ; ALU CARRY 15 L
5707
5708 002140  RBYTE0:    .WORD   377
5709 002142  RBYTE1:    .WORD   177400
5710 002144             ENDDAT
                        ;-----------------------------------------------------------------------------
```

```
                                                                    M 11
ZZ-ESKAD-13.2   VAX 11/780 MICRO DIAGNOSTIC HAR MACRO M1200  1  16-DEC-82          Fiche 1  Frame M11        Sequence 142
VAX 11/780 MICRO DIAGNOSTIC HAR MACRO M1200  16-DEC-82 09:58  PAGE 54-6
T30     DATA PATH ALU ARITHMETIC MODE

   5711
   5712
   5713 002144            FORCEOVERLAY
                .SBTTL    SECTION NUMBER OF
        000000             .PSECT  OVR17
   5714
```

```
5746                    .SBTTL  T31     DATA PATH ALU ARITHMETIC MODE
                        ;;*****************************************************************************
                        ;++
                        ;TEST   31      DATA PATH ALU ARITHMETIC MODE
                        ;
                        ; TEST DESCRIPTION
                        ;       THIS TEST CHECKS THE A+B, AND A-B, FUNCTIONS OF THE ALU. THE
                        ;       A+B FUNCTION IS TESTED WITH 16 DATA PATTERNS TO CHECK THE LOOK-
                        ;       AHEAD CARRY LOGIC ON THE ALU.
                        ;
                        ;       THIS IS DONE BY LOADING THE BLEG DATA INTO THE D REGISTER AND THE
                        ;       ALEG DATA INTO THE Q REGISTER, EXECUTING THE ALU FUNCTION AND PUTTING
                        ;       THE RESULT IN THE D REGISTER WHERE IT IS READ AND CHECKED.
                        ;
                        ;       SUBTST 1 - CHECK 16 DATA PATTERNS IN A+B MODE.
                        ;       SUBTST 2 - CHECK FOR NO CARRIES WHEN ALL THE P'S ARE HIGH AND
                        ;                  ALL THE G'S ARE LOW.
                        ;       SUBTST 3 - CHECK A-B OF ALTERNATING ONE'S AND ZERO'S.
                        ;
                        ; LOGIC DESCRIPTION
                        ;       THIS TEST CHECKS THE ALU FUNCTION CONTROL LOGIC ON THE DAP MODULE AND
                        ;       THE LOOK AHEAD CARRY LOGIC FOR THE ALU ON THE DCP, DDP, AND DEP MODULES.
                        ;
                        ; ERROR DESCRIPTION
                        ;
                        ;       IF THE OUTPUT DATA IS 4 DIGITS IN LENGTH, IT REPRESENTS THE V BUS
                        ;       SIGNAL FOR "ALU CARRY 31 L". IF IT IS 8 DIGITS IN LENGTH IT REPRESENTS
                        ;       THE ALU OUTPUT.
                        ;
                        ;       DATA: EXPECTED
                        ;             RECEIVED
                        ;             LOOP COUNT -- INDICATES WHICH DATA PATTERN IS BEING USED.
                        ;--
                        ;;*****************************************************************************
    000034              T31:
5750 000040                     INITIALIZE
5751 000042                     BLKMIC  XTMP70,,10000,3 ; LOAD THE MICRO WORDS TO DO AN A+B
5752
5753 000056                     SUBTEST
                        ;//////////////////////////////////////////////////////////////////////////////
    000056              T31S1:
5754                    ;+
5755                    ; CHECK 16 PATTERNS FOR BYTE 0
5756                    ;-
5757
5758 000060                     LOOP    I,1,16          ; CHECK THE FIRST 16 PATTERNS
5759 000072                     ERLOOP
5760 000074                     INITIALIZE
5761 000076                     LDIDREG RWDQ,XT2FL2,I   ; LOAD THE BLEG DATA
5762 000104                     FETCH   10000,,NONOP
5763 000114                     CLOCK   4               ; PUT IT IN THE D REGISTER
5764 000120                     LDIDREG RWDQ,XT2FL3,I   ; LOAD THE ALEG TEST DATA
5765 000126                     CLOCK   2               ; EXECUTE THE A+B
5766 000132                     TSTVB   XT2FV3          ; CHECK "ALU CARRY 31" ACTIVE
5767 000140                     IFERROR ,DP6D
5768 000146                     CLOCK   2               ; FINISH THE ALU FUNCTION
5769 000152                     READID  RWDQ            ; READ THE RESULT
```

```
                        CMPCAD  EQ,IDREGLO,XTMP73 ; CHECK THE RESULT
                        IFERROR 114,DP6E         ; ALU A+B FAILED
                        ENDLOOP I                ; CONTINUE
                        SKIP    SUBTST

                DP6D:   REPORT  <DCP>

                DP6E:   CMPCAM  ,IDREGLO,UBYTE0,,XTMP73 ; BYTE 0 OK?
                        CHKPNT  1$               ; BRANCH IF YES
                        CMPCAM  ,IDREGLO,UBYTE1,,XTMP73 ; BYTE 1 OK?
                        CHKPNT  2$               ; BRANCH IF YES
                        CMPCA   ,IDREGHI,XTMP73+2 ; BYTES 2,3 OK?
                        CHKPNT  3$               ; BRANCH IF YES
                        REPORT  <DAP>            ; ALL BYTES BAD
                2$:     REPORT  <DCP>            ; BYTE 0 BAD, BYTE 1 GOOD
                3$:     REPORT  <DCP,DDP>        ; BYTES 0,1 BAD, BYTES 2,3 GOOD

                1$:     CMPCAM  ,IDREGLO,UBYTE1,,XTMP73 ; BYTES 1 BAD?
                        CHKPNT  4$               ; BRANCH IF NO
                        REPORT  <DDP,DCP>        ; BYTE 0 GOOD, BYTE 1 BAD
                4$:     CMPCA   ,IDREGHI,XTMP73+2 ; BYTES 2,3 BAD?
                        CHKPNT  5$               ; BRANCH IF NO
                        REPORT  <DCP,DEP>        ; BYTES 0,1 GOOD, BYTES 2,3 BAD
                5$:     REPORT  <DAP,DCP,DDP,DEP> ; ALL BYTES GOOD


                        SUBTEST
                ;;;
```

```
                        INITIALIZE
                        LDIDREG RWDQ,XTMP17,I   ; LOAD THE BLEG TEST DATA
                        FETCH   10000,,NONOP
                        CLOCK   4               ; PUT B DATA IN D REG
                                                ; SAME DATA IN Q REG (A LEG)
                        CLOCK   4               ; EXECUTE THE NOP
SYNC7C:                 CLOCK   4               ; EXECUTE THE A-B
                        READID  RWDQ            ; GET THE RESULT
                        CMPCAD  EQ,IDREGLO,XTMP73 ; CHECK THE RESULT
                        IFERROR 117,DP6E        ; A-B FUNCTION FAILED
                        ENDLOOP I
                        SKIP

                        TESTDAT
                        ;-----------------------------------------------------------------------------
XTMP70:
                        ;*1000: DK/Q,J/1001             ; LOAD D WITH Q THRU DAL
                        .WORD   10001
                        .WORD   0
                        .WORD   4000
                        .WORD   600
                        .WORD   000074
                        .WORD   6000

                        ;*1001: BEN/0,J/1002            ; NOP
                        .WORD   10002
                        .WORD   0
                        .WORD   4000
                        .WORD   600
                        .WORD   000074
                        .WORD   0

                        ;*1002: RAMX/Q,RBMX/D,AMX/RAMX,BMX/RBMX,ALU/A+B,SHF/ALU,DK/SHF,J/1002
                        .WORD   10002
                        .WORD   0
                        .WORD   4000
                        .WORD   600
                        .WORD   20024
                        .WORD   4035

XTMP106:
                        ;*1002: RAMX/Q,RBMX/D,AMX/RAMX,BMX/RBMX,ALU/A-B,SHF/ALU,DK/SHF,J/1002
                        .WORD   10002
                        .WORD   0
                        .WORD   4000
                        .WORD   600
                        .WORD   20000
                        .WORD   4035

XTMP73:  .WORD   0,0


                        ;+
                        ; FOLLOWING IS THE BLEG DATA FOR SUBTEST 1
                        ;-

XT2FL2:  .WORD   125200,125252   ; HEX AA80,AAAA
```

```
5879 000766            .WORD   52500,52525     ; HEX 5540,5555
5880 000772            .WORD   125240,125252   ; HEX AAA0,AAAA
5881 000776            .WORD   52520,52525     ; HEX 5550,5555
5882 001002            .WORD   125250,125252   ; HEX AAA8,AAAA
5883 001006            .WORD   52524,52525     ; HEX 5554,5555
5884 001012            .WORD   125252,125252   ; HEX AAAA,AAAA
5885 001016            .WORD   52525,52525     ; HEX 5555,5555
5886 001022            .WORD   52600,52525     ; HEX 5580,5555
5887 001026            .WORD   125300,125252   ; HEX AAC0,AAAA
5888 001032            .WORD   52540,52525     ; HEX 5560,5555
5889 001036            .WORD   125260,125252   ; HEX AAB0,AAAA
5890 001042            .WORD   52530,52525     ; HEX 5558,5555
5891 001046            .WORD   125254,125252   ; HEX AAAC,AAAA
5892 001052            .WORD   52526,52525     ; HEX 5556,5555
5893 001056            .WORD   125253,125252   ; HEX AAAB,AAAA
5894
5895
5896            ;+
5897            ; FOLLOWING IS THE A LEG DATA FOR SUBTEST 1
5898            ;-
5899
5900 001062  XT2FL3: .WORD   52600,52525     ; HEX 5580,5555
5901 001066            .WORD   125300,125252   ; HEX AAC0,AAAA
5902 001072            .WORD   52540,52525     ; HEX 5560,5555
5903 001076            .WORD   125260,125252   ; HEX AAB0,AAAA
5904 001102            .WORD   52530,52525     ; HEX 5558,5555
5905 001106            .WORD   125254,125252   ; HEX AAAC,AAAA
5906 001112            .WORD   52526,52525     ; HEX 5556,5555
5907 001116            .WORD   125253,125252   ; HEX AAAB,AAAA
5908 001122            .WORD   125200,125252   ; HEX AA80,AAAA
5909 001126            .WORD   52500,52525     ; HEX 5540,5555
5910 001132            .WORD   125240,125252   ; HEX AAA0,AAAA
5911 001136            .WORD   52520,52525     ; HEX 5550,5555
5912 001142            .WORD   125250,125252   ; HEX AAA8,AAAA
5913 001146            .WORD   52524,52525     ; HEX 5554,5555
5914 001152            .WORD   125252,125252   ; HEX AAAA,AAAA
5915 001156            .WORD   52525,52525     ; HEX 5555,5555
5916            ;+
5917            ; FOLLOWING IS THE ALEG AND BLEG DATA FOR SUBTEST 3
5918            ;-
5919
5920 001162  XTMP17: .WORD   125252,125252,52525,52525,-1,-1
5921            ;   HEXDATA     AAAA,  AAAA,  5555, 5555,FFFF,FFFF
5922
5923 001176  XT2FL1: .WORD   SBC
5924 001200  XT2FL4: .WORD   42104,42104     ; HEX 4444,4444
5925 001204  XT2FL5: .WORD   104210,104210   ; HEX 8888,8888
5926 001210  XT2FL6: .WORD   1
5927 001212  XT2FV1: .WORD   1
5928 001214            VBUSG   1,23,1          ; ALU CARRY 07 L
5929 001216  XT2FV2: .WORD   1
5930 001220            VBUSG   1,22,1          ; ALU CARRY 15 L
5931 001222  XT2FV3: .WORD   1
5932 001224            VBUSG   1,21,0          ; ALU CARRY 31 L
5933 001226  XT2FV4: .WORD   1
5934 001230            VBUSG   1,22,0          ; ALU CARRY 15 L
5935
```

T31   DATA PATH ALU ARITHMETIC MODE

```
   5936 001232  UBYTE0: .WORD   377
   5937 001234  UBYTE1: .WORD   177400
   5938 001236          ENDDAT
                ;-------------------------------------------------------------------------------
   5939
   5940 001236          FORCEOVERLAY
                .SBTTL  SECTION NUMBER 10
        000000          .PSECT  OVR20
   5941
   5972
```

```
5976            .SBTTL  T32     ALU CARRY AND MODE CONTROL
                ;;*****************************************************************************
                ;++
                ;TEST   32      ALU CARRY AND MODE CONTROL
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THE ALU CARRY IN (TO BIT 0) AND THE ALU MODE
                ;       CONTROL. THIS IS DONE WITH 6 SUBTESTS, EACH SUBTEST EXECUTING
                ;       A DIFFERANT ALU FUNCTION. FOLLOWING IS A DESCRIPTION OF THE
                ;       FUNCTION, THE ALEG (D REG) AND BLEG (Q REG) DATA, THE EXPECTED
                ;       ALU OUTPUT, AND THE ALU OUTPUT IF THE FUNCTION FAILS:
                ;
                ;       SUBTEST ALU FUNCTION    ALEG    BLEG    EXP RES FAILURE RES
                ;       ------- ------------    ----    ----    ------- -----------
                ;         1     A - B - 1        0       0       -1        0
                ;         2     A + B + 1       1..1     0      1.....2   0 OR 1.....1
                ;         3     A - B[RLOG]     2..2    1..1    1.....1   3...3 OR 0
                ;         4     A + B + PSLC      0      -1       -1        0
                ;         5     A + B[RLOG]       0      -1       -1        0
                ;         6     A.AND.B         1..1     0        0       2....2
                ;
                ;       NOTE: 1...1 MEANS THERE IS A 1 IN EVERY NIBBLE (4 BITS).
                ;
                ;       THE MICRO CODE ROUTINE THAT IS LOADED TO EXECUTE THE FUNCTION
                ;       CONTAINS "NOP'S" FOR THOSE STATES WHEN THE CONSOLE IS LOADING
                ;       THE Q REGISTER WITH A DATA PATTERN.
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE LOGIC ON THE DAP MODULE THAT GENERATES THE
                ;       ALU CIN AND ALU MODE SIGNAL.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED ALU OUTPUT
                ;             RECEIVED ALU OUTPUT
                ;--
                ;;*****************************************************************************
    000034      T32:
5980 000040             INITIALIZE
5981
5982 000042             SUBTEST
                ;//////////////////////////////////////////////////////////////////////////////
    000042      T32S1:
5983            ;+
5984            ; A .MINUS. B .MINUS. 1
5985            ;-
5986
5987 000044             BLKMIC  T2AM1,,10000,5  ; LOAD THE MICRO CODE ROUTINE
5988 000060             ERLOOP
5989 000062             INITIALIZE
5990 000064             FETCH   10000,,NONOP    ; START AT LOCATION 1000(X)
5991 000074             LDIDREG RWDQ,T2AD1      ; LOAD THE ALEG DATA
5992 000102             CLOCK   4               ; PUT IT IN THE D REGISTER
5993 000106             LDIDREG RWDQ,T2AD1      ; LOAD THE BLEG DATA
5994 000114             CLOCK   4               ; EXECUTE THE ALU FUNCTION
5995 000120             READID  RWDQ            ; GET THE RESULT
5996 000124             CMPCAD  ,IDREGLO,T2AD2  ; CHECK RESULT = -1
5997 000150             IFERROR ,DAPCTL         ; ALU CONTROL FAILED.
```

```
                        SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
        000156  T32S2:
                ; CHECK THE A .PLUS. B .PLUS. 1 FUNCTION
                ;-

                        INITIALIZE
        000160          BLKMIC  T2AM2,,10003,1  ; LOAD MICRO CODE TO EXECUTE TEH FUNCTION
        000162          ERLOOP
        000176          INITIALIZE
        000200          FETCH   10000,,NONOP    ; START THE MICRO CODE
        000202          LDIDREG RWDQ,T2AD3      ; LOAD THE ALEG DATA
        000212          CLOCK   4               ; PUT IT IN D REG
        000220          LDIDREG RWDQ,T2AD1      ; LOAD THE B LEG DATA
        000224          CLOCK   4               ; EXECUTE THE FUNCTION
        000232          READID  RWDQ            ; READ THE RESULT
        000236          CMPCAD  ,IDREGLO,T2AD4  ; CHECK RESULT = 11111112
        000242          IFERROR ,DAPCTL

        000266

                        SUBTEST
        000274  ;////////////////////////////////////////////////////////////////////////////////
        000274  T32S3:
                ;+
                ; CHECK THE A .MINUS. B [RLOG] FUNCTION
                ;-

                        INITIALIZE
        000276          BLKMIC  T2AM3,,10003,1  ; LOAD THE MICRO CODE
        000300          ERLOOP
        000314          INITIALIZE
        000316          FETCH   10000,,NONOP    ; START THE MICRO CODE
        000320          LDIDREG RWDQ,T2AD5      ; LOAD THE A LEG DATA
        000330          CLOCK   4               ; PUT IT IN THE D REG
        000336          LDIDREG RWDQ,T2AD3      ; LOAD THE B LEG DATA
        000342          CLOCK   4               ; EXECUTE THE FUNCTION
        000350          READID  RWDQ            ; READ THE RESULT
        000354          CMPCAD  ,IDREGLO,T2AD3  ; CHECK THAT IT = 11111111
        000360          IFERROR ,DAPCTL

        000404

                        SUBTEST
        000412  ;////////////////////////////////////////////////////////////////////////////////
        000412  T32S4:
                ;+
                ; CHECK THE A .PLUS. B .PLUS. PSLC FUNCTION
                ;-

                        INITIALIZE
        000414          BLKMIC  T2AM4,,10003,1  ; LOAD THE MICRO WORDS
        000416          ERLOOP
        000432          INITIALIZE
        000434          LDIDREG PSL,T2AD1       ; ENSURE THE C BIT IS CLEAR
        000436          FETCH   10000,,NONOP    ; START THE MICRO CODE
        000444          LDIDREG RWDQ,T2AD1      ; LOAD THE A LEG DATA
        000454
```

```
6049 000462            CLOCK   4                   ; PUT IT IN THE D REG
6050 000466            LDIDREG RWDQ,T2AD2          ; LOAD THE B LEG DATA
6051 000474            CLOCK   4                   ; EXECUTE THE ALU FUNCTION
6052 000500            READID  RWDQ                ; READ THE RESULT
6053 000504            CMPCAD  ,IDREGLO,T2AD2      ; CHECK RESULT FOR -1
6054 000530            IFERROR ,DAPCTL             ;
6055
6056
6057 000536            SUBTEST
                    ;////////////////////////////////////////////////////////////////////////////////
     000536         T32S5:
6058                ;+
6059                ; CHECK THE A .PLUS. B [RLOG] FUNCTION
6060                ;-
6061
6062 000540            INITIALIZE
6063 000542            BLKMIC  T2AM5,,10003,1      ; LOAD THE MICRO WORDS
6064 000556            ERLOOP
6065 000560            INITIALIZE
6066 000562            FETCH   10000,,NONOP        ; START THE MICRO CODE
6067 000572            LDIDREG RWDQ,T2AD1          ; LOAD THE A LEG DATA
6068 000600            CLOCK   4                   ; PUT IT IN THE D REG
6069 000604            LDIDREG RWDQ,T2AD2          ; LOAD THE B LEG DATA
6070 000612            CLOCK   4                   ; EXECUTE THE FUNCTION
6071 000616            READID  RWDQ                ; READ THE RESULT
6072 000622            CMPCAD  ,IDREGLO,T2AD2      ; CHECK RESULT = -1
6073 000646            IFERROR ,DAPCTL
6074
6075
6076 000654            SUBTEST
                    ;////////////////////////////////////////////////////////////////////////////////
     000654         T32S6:
6077                ;+
6078                ; CHECK THE A .AND. B FUNCTION
6079                ;-
6080
6081 000656            INITIALIZE
6082 000660            BLKMIC  T2AM6,,10003,1      ; LOAD THE MICRO WORDS
6083 000674            ERLOOP
6084 000676            INITIALIZE
6085 000700            FETCH   10000,,NONOP        ; START EXECUTION
6086 000710            LDIDREG RWDQ,T2AD3          ; LOAD THE A LEG DATA
6087 000716            CLOCK   4                   ; PUT IT IN THE D REG
6088 000722            LDIDREG RWDQ,T2AD1          ; LOAD THE B LEG DATA
6089 000730            CLOCK   4                   ; EXECUTE THE FUNCTION
6090 000734            READID  RWDQ                ; READ THE RESULT
6091 000740            CMPCAD  ,IDREGLO,T2AD1      ; CHECK RESULT = 0
6092 000764            IFERROR ,DAPCTL
6093 000772            SKIP
6094
6095 000776  DAPCTL:  REPORT  <DAP>                ; ALU CONTROL LOGIC FAILED
6096
6097
6098 001006            TESTDAT
                    ;-------------------------------------------------------------------------------
6099 001012  T2AD1:   .WORD   0,0                  ; 32 BITS OF ZERO
6100 001016  T2AD2:   .WORD   -1,-1                ; 32 BITS OF ONE'S
```

```
6101 001022  T2AD3:  .WORD   10421,10421     ; 8 NIBBLES = TO 1
6102 001026  T2AD4:  .WORD   10422,10421     ; 8 NIBBLES = 11111112
6103 001032  T2AD5:  .WORD   21042,21042     ; 8 NIBBLES = TO 2
6104
6105 001036  T2AM1:
6106         ;1000:  NOP,J/1001
6107         ;1001:  D_Q,J/1002
6108         ;1002:  NOP,J/1003
6109         ;1003:  D_D-Q-1,J/1004
6110         ;1004:  NOP,J/1004
6111 001036          .WORD   10001
6112 001040          .WORD   0
6113 001042          .WORD   4000
6114 001044          .WORD   600
6115 001046          .WORD   74
6116 001050          .WORD   0
6117
6118 001052          .WORD   10002
6119 001054          .WORD   0
6120 001056          .WORD   4000
6121 001060          .WORD   600
6122 001062          .WORD   74
6123 001064          .WORD   6000
6124
6125 001066          .WORD   10003
6126 001070          .WORD   0
6127 001072          .WORD   4000
6128 001074          .WORD   600
6129 001076          .WORD   74
6130 001100          .WORD   0
6131
6132 001102          .WORD   10004
6133 001104          .WORD   0
6134 001106          .WORD   4000
6135 001110          .WORD   600
6136 001112          .WORD   10
6137 001114          .WORD   4035
6138
6139 001116          .WORD   10004
6140 001120          .WORD   0
6141 001122          .WORD   4000
6142 001124          .WORD   600
6143 001126          .WORD   74
6144 001130          .WORD   0
6145
6146 001132  T2AM2:
6147         ;1003:  D_D+Q+1,J/1004
6148 001132          .WORD   10004
6149 001134          .WORD   0
6150 001136          .WORD   4000
6151 001140          .WORD   600
6152 001142          .WORD   20
6153 001144          .WORD   4035
6154
6155 001146  T2AM3:
6156         ;1003:  RAMX/D,RBMX/Q,AMX/RAMX,BMX/RBMX,ALU/A-B.RLOG,D_ALU,J/1004
6157 001146          .WORD   10004
```

```
6158 001150            .WORD   0
6159 001152            .WORD   4000
6160 001154            .WORD   600
6161 001156            .WORD   4
6162 001160            .WORD   4035
6163
6164 001162  T2AM4:
6165                   ;1003:  RAMX/D,RBMX/Q,AMX/RAMX,BMX/RBMX,ALU/A+B+PSL.C,D_ALU,J/1004
6166 001162            .WORD   10004
6167 001164            .WORD   0
6168 001166            .WORD   4000
6169 001170            .WORD   600
6170 001172            .WORD   54
6171 001174            .WORD   4035
6172
6173 001176  T2AM5:
6174                   ;1003:  RAMX/D,RBMX/Q,AMX/RAMX,BMX/RBMX,ALU/A+B.RLOG,D_ALU,J/1004
6175 001176            .WORD   10004
6176 001200            .WORD   0
6177 001202            .WORD   4000
6178 001204            .WORD   600
6179 001206            .WORD   30
6180 001210            .WORD   4035
6181
6182 001212  T2AM6:
6183                   ;1003:  D_D.AND.Q,J/1004
6184 001212            .WORD   10004
6185 001214            .WORD   0
6186 001216            .WORD   4000
6187 001220            .WORD   600
6188 001222            .WORD   64
6189 001224            .WORD   4035
6190 001226            ENDDAT
                       ;-----------------------------------------------------------------------------
6191
```

```
                .SBTTL  T33     DATA PATH KMX
                ;;*****************************************************************************
                ;++
                ;TEST   33      DATA PATH KMX
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THAT THE FAST CONSTANTS
                ;       CAN BE READ THROUGH THE KMX.
                ;
                ;       THIS IS DONE BY LOADING A MICRO INSTRUCTION WHO'S KMX FIELD HAS THE
                ;       APPROPRIATE VALUE IN IT, AND TRANSFERING THE KMX TO THE D REGISTER
                ;       WHERE IT IS CHECKED.
                ;
                ;       SUBTST 1 - CHECK THE FAST CONSTANTS
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE LOGIC ON THE DAP MODULE THAT SELECTS THE KMUX AND BMUX
                ;       AND THE KMUX INPUT TO THE BMX
                ;       ON THE DDP MODULE.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED KMUX OUTPUT
                ;             RECEIVED KMUX OUTPUT
                ;             LOOP COUNT -- INDICATES WHICH ADDRESS OF THE KMX FIELD IS BE
                ;               SELECTED, I.E. 1=ADR 0, 2=ADR 1, 3=ADR, 2, ETC.
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC7D--FAST CONSTANT IS GATED THRU THE ALU
                ;--
                ;;*****************************************************************************
    001226      T33:
6223 001232             INITIALIZE
6224
6225 001234             SUBTEST
                ;/////////////////////////////////////////////////////////////////////////////
    001234      T33S1:
6226            ;+
6227            ; FIRST CHECK THE FAST CONSTANTS (EXCEPT FOR THE SP1 AND SP2)
6228            ;-
6229
6230 001236             LOOP    I,1,5
6231 001250             INITIALIZE
6232 001252             KMXGEN  TMP110,I         ; GENERATE KMX FIELD VALUE
6233 001260             BLKMIC  TMP110,,10000,1 ; LOAD MICRO WORD
6234 001274             ERLOOP
6235 001276             INITIALIZE
6236 001300             FETCH   10000,,NONOP
6237 001310     SYNC7D: CLOCK   4                ; PUT THE SELECTED CONSTANT IN THE D REG
6238 001314             READID  RWDQ             ; READ THE D REG
6239 001320             CMPCAD  EQ,IDREGLO,TMP111,I ; CHECK THE CONTENTS OF THE D REG
6240 001344             IFERROR 120,DP7          ; FAST CONSTANT MUX FAILED
6241 001352             ENDLOOP I                ; CONTINUE
6242 001356             SKIP
6243
6244
6245 001362     DP7:    MOVE    TMP111,I,TMP112 ; GET EXPECTED DATA
6246 001372             MASK    TMP112,SBYTE0   ; GET BYTE0 ONLY
```

```
 6247 001400            CMPCAM  ,IDREGLO,SBYTE0,,TMP112 ; IS BYTE 0 BAD?
 6248 001414            CHKPNT  1$              ; BRANCH IF NO
 6249 001422            MOVE    TMP111,I,TMP112 ; GET EXPECTED DATA AGAIN
 6250 001432            MASK    TMP112,SBYTE1   ; GET BYTE 1 ONLY
 6251 001440            CMPCAM  ,IDREGLO,SBYTE1,,TMP112 ; IS BYTE 1 BAD?
 6252 001454            CHKPNT  2$              ; BRANCH IF NO
 6253 001462            CMPCA   ,IDREGHI,TMP112+2 ; ARE BYTES 2,3 BAD?
 6254 001474            CHKPNT  3$              ; BRANCH IF NO
 6255 001502            REPORT  <DAP>           ; ALL BYTES BAD (CHECK BMUX SELECT)
 6256 001512  3$:       REPORT  <DCP,DDP>       ; BYTES 0 AND 1 BAD, BYTES 2,3 GOOD
 6257 001522  2$:       REPORT  <DCP>           ; BYTE 0 BAD AND BYTE 1 GOOD
 6258
 6259 001532  1$:       MOVE    TMP111,I,TMP112 ; GET EXPECTED DATA
 6260 001542            MASK    TMP112,SBYTE1   ; ONLY WANT BYTE 1
 6261 001550            CMPCAM  ,IDREGLO,SBYTE1,,TMP112 ; IS BYTE 1 BAD?
 6262 001564            CHKPNT  4$              ; BRANCH IF NO
 6263 001572            REPORT  <DDP>           ; BYTE 0 GOOD, BYTE 1 BAD
 6264 001602  4$:       CMPCA   ,IDREGHI,TMP112+2 ; BYTES 2,3 BAD?
 6265 001614            CHKPNT  5$              ; BRANCH IF NO
 6266 001622            REPORT  <DDP,DEP>       ; BYTE 0 GOOD, BYTES 1,2,3 BAD
 6267 001632  5$:       REPORT  <DAP,DCP,DDP,DEP> ; ALL BYTES GOOD (SHOULD NEVER HAPPEN)
 6268
 6269 001642            TESTDAT
                ;-------------------------------------------------------------------------------
 6270 001646  TMP110:
 6271           ;*1000: KMX/.8,BMX/KMX,ALU/B,SHF/ALU,DK/SHF,J/1000
 6272 001646            .WORD   10000
 6273 001650            .WORD   0
 6274 001652            .WORD   4000
 6275 001654            .WORD   600
 6276 001656            .WORD   000070
 6277 001660            .WORD   4030
 6278
 6279           ;+
 6280           ; FOLLOWING IS THE EXPECTED FAST CONSTANTS
 6281           ;-
 6282
 6283 001662  TMP111:  .WORD   8.,0
 6284 001666           .WORD   1.,0
 6285 001672           .WORD   2.,0
 6286 001676           .WORD   3.,0
 6287 001702           .WORD   4.,0
 6288
 6289 001706  TMP112:  .WORD   0,0
 6290 001712  T2BL1:   .WORD   SBC             ;
 6291 001714  SBYTE0:  .WORD   377
 6292 001716  SBYTE1:  .WORD   177400
 6293 001720           ENDDAT
                ;-------------------------------------------------------------------------------
 6294
```

```
                .SBTTL  T34     ID BUS WRITE/READ
                ;;*****************************************************************************
                ;++
                ;TEST   34      ID BUS WRITE/READ
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THAT THE MICRO CODE (ALONG WITH THE DATA PATH)
                ;       CAN READ AND WRITE THE CONSOLE INTERFACE ID BUS REGISTERS.
                ;
                ;       THIS IS DONE BY LOADING A MICRO INSTRUCTIONS THAT WRITES (OR READS)
                ;       THE ID BUS, PUTTING THE DATA PATTERN IN THE D REGISTER OR CONSOLE
                ;       TO ID REGISTER, EXECUTING THE MICRO INSTRUCTION, AND CHECKING THAT THE
                ;       DATA PATTERN WAS TRANSFERED CORRECTLY.
                ;
                ;       SUBTST 1 - CHECK AN ID BUS WRITE TO THE "FROM ID" EGISTER
                ;       SUBTST 2 - CHECK AN ID BUS READ FROM THE "TO ID" REGISTER
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE LOGIC ON THE DAP AND ICL MODULES THAT GENERATED
                ;       ID BUS READS AND WRITES WITH THE ADDRESS SELECTED BY THE KMX FIELD.
                ;       IT ALSO CHECKS THE Q REGISTER CLOCK CONTROL ON AN ID BUS READ.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED ID BUS DATA
                ;             RECEIVED ID BUS DATA
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC7F--D REGISTER SHOULD BE ENABLED ONTO THE ID BUS
                ;       SUBTST 2 - SYNC80--Q REGISTER SHOULD BE RECEIVING THE ID BUS DATA
                ;--
                ;;*****************************************************************************
    001720      T34:
6327 001724             INITIALIZE
6328
6329 001726             SUBTEST
                ;/////////////////////////////////////////////////////////////////////////////
    001726      T34S1:
6330            ;+
6331            ; FIRST CHECK THAT THE D REGISTER CAN BE WRITTEN INTO THE "FROM ID" REGISTER
6332            ;-
6333
6334 001730             BLKMIC  TMP113,,10000,2 ; LOAD MICRO WORDS
6335 001744             ERLOOP
6336 001746             INITIALIZE
6337 001750             FETCH   10000,,NONOP
6338 001760             CLOCK   4               ; LOAD D REGISTER WITH A CONSTANT
6339 001764             CLOCK   3               ; GO TO CPT150
6340 001770     SYNC7F: CLOCK   1               ; ID BUS IS ACTIVE, AND GO TO CPT0
6341 001774             CMPCAD  EQ,FMIDLO,TMP114 ; CHECK IF DATA WAS WRITTEN INTO FROM ID REG
6342 002020             IFERROR 122,DP10        ; ID BUS WRITE FAILED
6343
6344 002026             SUBTEST
                ;/////////////////////////////////////////////////////////////////////////////
    002026      T34S2:
6345            ;+
6346            ; NOW CHECK THE ID BUS READ FUNCTION
6347            ;-
```

```
6348
6349 002030            INITIALIZE                  ; SET ROM NOP
6350 002032            BLKMIC  TMP115,,10000,2     ; LOAD MICRO INSTRUCTIONS
6351 002046            ERLOOP
6352 002050            INITIALIZE
6353 002052            FETCH   10000,,NONOP
6354 002062            LOADCA  TOIDLO,TMP114       ; PUT DATA IN "TO ID" REGISTER
6355 002072            LOADCA  TOIDHI,TMP114+2     ; ...
6356 002102            CLOCK   3                   ; GO TO CPT150 OF THE READ
6357 002106  SYNC80:   CLOCK   1                   ; LOAD THE Q REGISTER WITH THE DATA
6358 002112            CLOCK   4                   ; PUT THE DATA IN THE D REGISTER
6359 002116            READID  RWDQ                ; READ THE D REG
6360 002122            CMPCAD  EQ,IDREGLO,TMP114   ; CHECK THAT TO ID REG WENT TO THE DATA PATH
6361 002146            IFERROR 123,DP10            ; READ ID BUS FAILED
6362 002154            SKIP
6363
6364 002160  DP10:     REPORT  <ICL,CIB>
6365
6366 002170            TESTDAT
                ;-------------------------------------------------------------------------------
6367 002174  TMP113:
6368            ;*1000: D K[.3030],J/1001
6369 002174            .WORD   10001
6370 002176            .WORD   0
6371 002200            .WORD   4000
6372 002202            .WORD   144600
6373 002204            .WORD   000070
6374 002206            .WORD   4030
6375
6376            ;*1001: ID[TXDB] D,J/1001
6377 002210            .WORD   10001
6378 002212            .WORD   0
6379 002214            .WORD   036000
6380 002216            .WORD   16600
6381 002220            .WORD   000074
6382 002222            .WORD   0
6383
6384 002224  TMP115:
6385            ;*1000: Q ID[RXDB],J/1001
6386 002224            .WORD   10001
6387 002226            .WORD   0
6388 002230            .WORD   126000
6389 002232            .WORD   12760
6390 002234            .WORD   000074
6391 002236            .WORD   0
6392            ;*1001: DK/Q,J/1001
6393 002240            .WORD   10001
6394 002242            .WORD   0
6395 002244            .WORD   4000
6396 002246            .WORD   600
6397 002250            .WORD   000074
6398 002252            .WORD   6000
6399
6400 002254  TMP114:   .WORD   30060,0
6401 002260  T2CL1:    .WORD   SBC
6402 002262            ENDDAT
                ;-------------------------------------------------------------------------------
```

T34 ID BUS WRITE/READ

6403

```
                .SBTTL  SECTION NUMBER 11
                .PSECT  OVR21
                .SBTTL  T35     "D BYTES" BRANCH
                ;;*****************************************************************************
                ;++
                ;TEST   35      "D BYTES" BRANCH
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST FLOAT A ONE THRU BYTES 0, 1, 2, AND 3 OF THE D REG
                ;       TO CHECK THE " BYTES" BRANCH. THE RESULT IS CHECKED BY LOOKING
                ;       AT THE UPCSV REGISTER AFTER THE BRANCH.
                ;
                ;       SUBTST 1 - CHECK BYTE 0
                ;       SUBTST 2 - CHECK BYTE 1
                ;       SUBTST 3 - CHECK BYTE 2
                ;       SUBTST 4 - CHECK BYTE 3
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE LOGIC ON THE DBP MODULE THAT GENERATES THE
                ;       D REG.NE.0 SIGNALS AND THE BRANCH MULTIPLEXORS ON THE DBP AND ICL MODULES.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED UPC BITS<12:0> AFTER THE BRANCH
                ;             RECEIVED UPC BITS<12:0> AFTER THE BRANCH
                ;             LOOP COUNT -- INDICATES THE BIT POSITION OF THE FLOATING ONE
                ;               OR ZERO, I.E. 1=BIT 0, 2=BIT 1, 3=BIT 2, ETC.
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC81--BRANCH BITS ARE ASSERTED ON THE NUA BUS
                ;       SUBTST 2 - SYNC82--BRANCH BITS ARE ASSERTED ON THE NUA BUS
                ;       SUBTST 3 - SYNC83--BRANCH BITS ARE ASSERTED ON THE NUA BUS
                ;       SUBTST 4 - SYNC84--BRANCH BITS ARE ASSERTED ON THE NUA BUS
                ;--
                ;;*****************************************************************************
    000034      T35:
6437 000040             INITIALIZE
6438
6439 000042             SUBTEST
                ;//////////////////////////////////////////////////////////////////////////////
    000042      T35S1:
6440            ;+
6441            ;       FIRST CHECK BYTE ZERO
6442            ;-
6443
6444 000044             BLKMIC  TMP116,,10000,2 ; LOAD MICRO INSTR TO LOAD REG AND ENABLE THE BRANCH
6445 000060             LOOP    I,1,8           ; CHECK BYTE 0
6446 000072             FLTONE  T2DL1,I         ; GENERATE THE TEST PATTERN
6447 000100             ERLOOP
6448 000102             INITIALIZE
6449 000104             LDIDREG RWDQ,T2DL1      ; LOAD TEST DATA INTO Q REG
6450 000112             FETCH   10001,,NONOP    ; START THE TEST
6451 000122             CLOCK   4               ; PUT THE PATTERN IN THE D REGISTER
6452 000126     SYNC81: CLOCK   4               ; EXECUTE THE BRANCH
6453 000132             CMPPCSV TMP117          ; CHECK RESULT OF THE BRANCH
6454 000140             IFERROR 124,DP11        ; "D BYTES" BRANCH FAILED
6455 000146             ENDLOOP I               ; CONTINUE
6456 000152             SKIP    SUBTST
```

T35 "D BYTES" BRANCH

```
                                                                                                                    
                                                                                                                    
000156  DP11:   INITIALIZE
000160          LDIDREG RWDQ,T2DL1      ; RELOAD THE DATA PATTERN
000166          FETCH   10001,,NONOP    ; RESTART EXECUTION
000176          CLOCK   4               ; LOAD THE D REGISTER
000202          CLOCK   2               ; STOP IN MIDDLE OF BRANCH INSTR
000206          TSTVB   T33V1           ; CHECK THE BEMX SIGNALS FROM ICL TO DBP
000214          CHKPNT  1$              ; BRANCH IF OK
000222          REPORT  <ICL>           ; BEMX IS INCORRECT
000232  1$:     TSTVB   T33V2           ; CHECK THE BR BIT FROM DBP TO ICL
000240          CHKPNT  2$              ; BRANCH IF OK
000246          REPORT  <DBP>           ; BRANCH BITS INCORRECT
000256  2$:     TSTVB   T33V3           ; CHECK THE BR BIT FROM ICL TO USC
000264          CHKPNT  3$              ; BRANCH IF OK
000272          REPORT  <ICL>           ; BR BIT INCORRECT FROM ICL TO USC
000302  3$:     REPORT  <USC>           ; BR BITS OK UPC INCORRECT
                                                                                                                    
                                                                                                                    
                                                                                                                    
000312          SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////////////////////
000312  T35S2:
                ;+
                ; NOW CHECK BYTE 1
                ;-
                                                                                                                    
000314          LOOP    I,9,16
000326          FLTONE  T2DL1,I         ; GENERATE THE TEST PATTERN
000334          ERLOOP
000336          INITIALIZE
000340          LDIDREG RWDQ,T2DL1      ; LOAD THE TEST PATTERN
000346          FETCH   10001,,NONOP    ; START THE TEST
000356          CLOCK   4               ; LOAD THE D REGISTER
000362  SYNC82: CLOCK   4               ; EXECUTE THE BRANCH
000366          CMPPCSV TMP120          ; CHECK THE RESULT OF THE BRANCH
000374          IFERROR 126,DP11A       ; "D BYTES" BRANCH FAILED
000402          ENDLOOP I               ; CONTINUE
000406          SKIP    SUBTST
                                                                                                                    
                                                                                                                    
000412  DP11A:  INITIALIZE
000414          LDIDREG RWDQ,T2DL1      ; RELOAD THE DATA PATTERN
000422          FETCH   10001,,NONOP    ; RESTART EXECUTION
000432          CLOCK   4               ; LOAD THE D REGISTER
000436          CLOCK   2               ; STOP IN MIDDLE OF BRANCH INSTR
000442          TSTVB   T33V1           ; CHECK THE BEMX SIGNALS FROM ICL TO DBP
000450          CHKPNT  1$              ; BRANCH IF OK
000456          REPORT  <ICL>           ; BEMX IS INCORRECT
000466  1$:     TSTVB   T33V2A          ; CHECK THE BR BIT FROM DBP TO ICL
000474          CHKPNT  2$              ; BRANCH IF OK
000502          REPORT  <DBP>           ; BRANCH BITS INCORRECT
000512  2$:     TSTVB   T33V3A          ; CHECK THE BR BIT FROM ICL TO USC
000520          CHKPNT  3$              ; BRANCH IF OK
000526          REPORT  <ICL>           ; BR BIT INCORRECT FROM ICL TO USC
000536  3$:     REPORT  <USC>           ; BR BITS OK UPC INCORRECT
                                                                                                                    
```

```
                        SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
        000546  T35S3:
                ;+
                ; NOW CHECK BYTE 2
                ;-

000550                  LOOP    I,17,24
000562                  FLTONE  T2DL1,I         ; GENERATE THE TEST PATTERN
000570                  ERLOOP
000572                  INITIALIZE
000574                  LDIDREG RWDQ,T2DL1      ; LOAD THE TEST PATTERN
000602                  FETCH   10001,,NONOP
000612                  CLOCK   4               ; PUT THE PATTERN IN THE D REG
000616  SYNC83:         CLOCK   4               ; EXECUTE THE BRANCH
000622                  CMPPCSV TMP121          ; CHECK THE RESULT OF THE BRANCH
000630                  IFERROR 125,DP11B       ; "D BYTES" BRANCH FAILED
000636                  ENDLOOP I               ; CONTINUE
000642                  SKIP    SUBTST


000646  DP11B:          INITIALIZE
000650                  LDIDREG RWDQ,T2DL1      ; RELOAD THE DATA PATTERN
000656                  FETCH   10001,,NONOP    ; RESTART EXECUTION
000666                  CLOCK   4               ; LOAD THE D REGISTER
000672                  CLOCK   2               ; STOP IN MIDDLE OF BRANCH INSTR
000676                  TSTVB   T33V1           ; CHECK THE BEMX SIGNALS FROM ICL TO DBP
000704                  CHKPNT  1$              ; BRANCH IF OK
000712                  REPORT  <ICL>           ; BEMX IS INCORRECT
000722  1$:             TSTVB   T33V2B          ; CHECK THE BR BIT FROM DBP TO ICL
000730                  CHKPNT  2$              ; BRANCH IF OK
000736                  REPORT  <DBP>           ; BRANCH BITS INCORRECT
000746  2$:             TSTVB   T33V3B          ; CHECK THE BR BIT FROM ICL TO USC
000754                  CHKPNT  3$              ; BRANCH IF OK
000762                  REPORT  <ICL>           ; BR BIT INCORRECT FROM ICL TO USC
000772  3$:             REPORT  <USC>           ; BR BITS OK UPC INCORRECT


001002                  SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
        001002  T35S4:
                ;+
                ; NOW CHECK BYTE 3
                ;-

001004                  LOOP    I,25,32
001016                  FLTONE  T2DL1,I         ; GENERATE THE TEST PATTERN
001024                  ERLOOP
001026                  INITIALIZE
001030                  LDIDREG RWDQ,T2DL1      ; LOAD THE TEST PATTERN
001036                  FETCH   10001,,NONOP
001046                  CLOCK   4               ; PUT THE PATTERN IN THE D REG
001052  SYNC84:         CLOCK   4               ; EXECUTE THE BRANCH
001056                  CMPPCSV TMP122          ; CHECK THE RESULT
```

```
6565 001064            IFERROR 127,DP11C       ; "D BYTES" BRANCH FAILED
6566 001072            ENDLOOP I               ; CONTINUE
6567 001076            SKIP
6568
6569
6570 001102  DP11C:    INITIALIZE
6571 001104            LDIDREG RWDQ,T2DL1      ; RELOAD THE DATA PATTERN
6572 001112            FETCH   10001,,NONOP    ; RESTART EXECUTION
6573 001122            CLOCK   4               ; LOAD THE D REGISTER
6574 001126            CLOCK   2               ; STOP IN MIDDLE OF BRANCH INSTR
6575 001132            TSTVB   T33V1           ; CHECK THE BEMX SIGNALS FROM ICL TO DBP
6576 001140            CHKPNT  1$              ; BRANCH IF OK
6577 001146            REPORT  <ICL>           ; BEMX IS INCORRECT
6578 001156  1$:       TSTVB   T33V2C          ; CHECK THE BR BIT FROM DBP TO ICL
6579 001164            CHKPNT  2$              ; BRANCH IF OK
6580 001172            REPORT  <DBP>           ; BRANCH BITS INCORRECT
6581 001202  2$:       TSTVB   T33V3C          ; CHECK THE BR BIT FROM ICL TO USC
6582 001210            CHKPNT  3$              ; BRANCH IF OK
6583 001216            REPORT  <ICL>           ; BR BIT INCORRECT FROM ICL TO USC
6584 001226  3$:       REPORT  <USC>           ; BR BITS OK UPC INCORRECT
6585
6586
6587 001236            TESTDAT
                ;-------------------------------------------------------------------------------
6588 001242  TMP116:
6589                   ;*1000: BEN/D.BYTES,J/1000              ; BRANCH ENABLE
6590 001242            .WORD   10000
6591 001244            .WORD   0
6592 001246            .WORD   4000
6593 001250            .WORD   600
6594 001252            .WORD   014074
6595 001254            .WORD   0
6596
6597                   ;*1001: DK/Q,J/1000             ; LOAD D WITH TEST PATTERN
6598 001256            .WORD   10000
6599 001260            .WORD   0
6600 001262            .WORD   4000
6601 001264            .WORD   600
6602 001266            .WORD   000074
6603 001270            .WORD   6000
6604
6605                   ;+
6606                   ; FOLLOWING IS THE EXPECTED UPC BITS<12:0> FOR THE 4 SUBTESTS
6607                   ;-
6608
6609 001272  TMP117:   .WORD   10001           ; BRANCH ADDRESS D.BYTES AND BYTE 0
6610 001274  TMP120:   .WORD   10002           ; BRANCH ADDRESS D.BYTES AND BYTE 1
6611 001276  TMP121:   .WORD   10004           ; BRANCH ADDRESS D.BYTES AND BYTE 2
6612 001300  TMP122:   .WORD   10010           ; BRANCH ADDRESS D.BYTES AND BYTE 3
6613 001302  T2DL0:    .WORD   10000           ; BRANCH ADDRESS D REG = 0
6614
6615 001304  T2DL1:    .WORD   0,0
6616 001310  T2DL2:    .WORD   SBC
6617 001312  T2DL3:    .WORD   0,0
6618
6619 001316  T33V1:    .WORD   3               ; BEXM SIGNALS FROM ICL TO DBP
6620 001320            VBUSG   2,122,1         ; REM BEMX S2 H
```

```
6621 001322            VBUSG   2,123,0          ; REM BEMX S1 H
6622 001324            VBUSG   2,124,0          ; REM BEMX S0 H
6623 001326  T33V2:    .WORD   4                ; BR BIT SIGNALS FROM DBP TO ICL
6624 001330            VBUSG   2,114,1          ; BR BIT0 H
6625 001332            VBUSG   2,112,0          ; BR BIT1 H
6626 001334            VBUSG   2,110,0          ; BR BIT2 H
6627 001336            VBUSG   2,106,0          ; BR BIT3 H
6628 001340  T33V2A:   .WORD   4
6629 001342            VBUSG   2,114,0          ; BR BIT0 H
6630 001344            VBUSG   2,112,1          ; BR BIT1 H
6631 001346            VBUSG   2,110,0          ; BR BIT2 H
6632 001350            VBUSG   2,106,0          ; BR BIT3 H
6633 001352  T33V2B:   .WORD   4
6634 001354            VBUSG   2,114,0          ; BR BIT0 H
6635 001356            VBUSG   2,112,0          ; BR BIT1 H
6636 001360            VBUSG   2,110,1          ; BR BIT2 H
6637 001362            VBUSG   2,106,0          ; BR BIT3 H
6638 001364  T33V2C:   .WORD   4
6639 001366            VBUSG   2,114,0          ; BR BIT0 H
6640 001370            VBUSG   2,112,0          ; BR BIT1 H
6641 001372            VBUSG   2,110,0          ; BR BIT2 H
6642 001374            VBUSG   2,106,1          ; BR BIT3 H
6643
6644 001376  T33V3:    .WORD   4
6645 001400            VBUSG   0,103,1          ; BRBIT0(1B:14) H
6646 001402            VBUSG   0,106,0          ; BRBIT1(1B:14) H
6647 001404            VBUSG   0,111,0          ; BRBIT2(1B:14) H
6648 001406            VBUSG   0,114,0          ; BRBIT3(1B:14) H
6649 001410  T33V3A:   .WORD   4
6650 001412            VBUSG   0,103,0          ; BRBIT0(1B:14) H
6651 001414            VBUSG   0,106,1          ; BRBIT1(1B:14) H
6652 001416            VBUSG   0,111,0          ; BRBIT2(1B:14) H
6653 001420            VBUSG   0,114,0          ; BRBIT3(1B:14) H
6654 001422  T33V3B:   .WORD   4
6655 001424            VBUSG   0,103,0          ; BRBIT0(1B:14) H
6656 001426            VBUSG   0,106,0          ; BRBIT0(1B:14) H
6657 001430            VBUSG   0,111,1          ; BRBIT2(1B:14) H
6658 001432            VBUSG   0,114,0          ; BRBIT3(1B:14) H
6659 001434  T33V3C:   .WORD   4
6660 001436            VBUSG   0,103,0          ; BRBIT0(1B:14) H
6661 001440            VBUSG   0,106,0          ; BRBIT1(1B:14) H
6662 001442            VBUSG   0,111,0          ; BRBIT2(1B:14) H
6663 001444            VBUSG   0,114,1          ; BRBIT3(1B:14) H
6664 001446            ENDDAT
                ;-------------------------------------------------------------------------------
6665
6666 001446            FORCEOVERLAY
                .SBTTL  SECTION NUMBER 12
     000000            .PSECT  OVR22
6667
```

```
                .SBTTL  T36     ALU BRANCH
                ;;*****************************************************************************
                ;++
                ;TEST   36      ALU BRANCH
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THE ALU ZERO (BEN 1) BRANCH BY FLOATING A ONE THRU
                ;       EACH BYTE OF THE ALU AND ENSURING THE BRANCH DOES NOT ENABLE, AND
                ;       THEN SETTING THE BYTE TO ALL ZERO'S AND CHECKING THAT THE BRANCH DOES
                ;       ENABLE, WITH THE 3 DATA TYPES (BYTE, WORD, AND LONG).
                ;
                ;       SUBTST 1 - CHECK BYTE 0 - DATA TYPE BYTE
                ;       SUBTST 2 - CHECK BYTE 0 AND 1 - DATA TYPE WORD
                ;       SUBTST 3 - CHECK BYTES 0 THRU 3 - DATA TYPE LONG
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE LOGIC THAT GENERATES THE ALU = 0 SIGNALS ON THE
                ;       DBP, DCP, DDP, AND CEH MODULES AND THE BRANCH MULTIPLEXORS ON THE
                ;       CEH AND USC MODULES.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED UPC BITS<12:0>
                ;             RECEIVED UPC BITS<12:0>
                ;             LOOP COUNT -- INDICATES THE BIT POSITION OF THE FLOATING ONE,
                ;               I.E. 1=BIT 0, 2=BIT 1, 3=BIT 2, ETC.
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC85--BRANCH BITS ARE ACTIVE
                ;                  SYNC86--BRANCH BITS ARE ACTIVE (SHOULD BE ASSERTED)
                ;       SUBTST 2 - SYNC87--BRANCH BITS ARE ACTIVE (DEASSERTED)
                ;                  SYNC88--BRANCH BITS ARE ACTIVE (ASSERTED)
                ;       SUBTST 3 - SYNC89--BRANCH BITS ARE ACTIVE (DEASSERTED)
                ;                  SYNC8A--BRANCH BITS ARE ACTIVE (ASSERTED)
                ;--
                ;;*****************************************************************************
    000034      T36:
6704 000040             INITIALIZE
6705
6706 000042             SUBTEST
                ;//////////////////////////////////////////////////////////////////////////////
    000042      T36S1:
6707            ;+
6708            ; FIRST CHECK DATA TYPE BYTE
6709            ;-
6710
6711 000044             BLKMIC  TMP123,,10000,2 ; LOAD THE SETUP AND BRANCH MICROWORDS
6712 000060             LOOP    I,1,8           ; GOING TO CHECK DATA TYPE BYTE
6713 000072             FLTONE  TMP130,I        ; GENERATE THE TEST PATTERN
6714 000100             ERLOOP
6715 000102             INITIALIZE
6716 000104             LDIDREG RWDQ,TMP130     ; LOAD THE DATA PATTERN
6717 000112             FETCH   10000,,NONOP
6718 000122             CLOCK   4               ; LATCH THE MICRO CONDITION CODES
6719 000126     SYNC85: CLOCK   4               ; EXECUTE THE BRANCH
6720 000132             CMPPCSV XTMP126         ; CHECK FOR NO BRANCH
6721 000140             IFERROR 127,DP12        ; ALU ZERO BRANCH FAILED, DT=BYTE
6722 000146             ENDLOOP I               ; CONTINUE
```

```
6723
6724                    ;+
6725                    ; NOW CHECK THAT BRANCH BIT ASCERTS
6726                    ;-
6727
6728 000152                    LDIDREG RWDQ,TMP126     ; PUT ZERO'S IN Q REG
6729 000160                    ERLOOP
6730 000162                    INITIALIZE
6731 000164                    FETCH   10000,,NONOP
6732 000174                    CLOCK   4               ; LATCH THE MICRO CONDITION CODES
6733 000200  SYNC86:           CLOCK   4               ; EXECUTE THE BRANCH
6734 000204                    CMPPCSV TMP127          ; CHECK THAT BRANCH ASCERTED
6735 000212                    IFERROR 131,DP12A       ; ALU ZERO BRANCH FAILED, DT=BYTE
6736 000220                    SKIP    SUBTST
6737
6738 000224  DP12:            INITIALIZE
6739 000226                    LDIDREG RWDQ,TMP130     ; LOAD TEST PATTERN
6740 000234                    FETCH   10000,,NONOP    ; START THE SEQUENCER
6741 000244                    CLOCK   2               ; ENABLE THE ALU DATA
6742 000250                    TSTVB   T34V1           ; CHECK "ALU (7:0)=0" INACTIVE
6743 000256                    CHKPNT  1$              ; BRANCH IF OK
6744 000264                    REPORT  <DCP>           ; ALU (7:0)=0 IS ACTIVE
6745 000274  1$:               CLOCK   2               ; COMPLETE THE INSTRUCTION
6746 000300                    TSTVB   T34V2           ; CHECK "ALU Z(1)" INACTIVE
6747 000306                    CHKPNT  2$              ; BRANCH IF OK
6748 000314                    REPORT  <CEH,ICL>       ; ICL ALU Z(1) INCORRECT
6749 000324  2$:               REPORT  <USC>           ; ALU Z(1) OK BUT BRANCH FAILED
6750
6751 000334  DP12A:           INITIALIZE
6752 000336                    LDIDREG RWDQ,TMP126     ; LOAD Q REG WITH ZEROS
6753 000344                    FETCH   10000,,NONOP    ; START THE SEQUENCER
6754 000354                    CLOCK   2               ; ENABLE THE ALU DATA
6755 000360                    TSTVB   T34V1A          ; CHECK "ALU (7:0)=0" ACTIVE
6756 000366                    CHKPNT  1$              ; BRANCH IF OK
6757 000374                    REPORT  <DCP>           ; ALU (7:0)=0 FAILED
6758 000404  1$:               CLOCK   2               ; COMPLETE THE INSTRUCTION
6759 000410                    TSTVB   T34V2A          ; CHECK "ALU Z(1)" ACTIVE
6760 000416                    CHKPNT  2$              ; BRANCH IF OK
6761 000424                    REPORT  <CEH,ICL>       ; MICRO CONDITION CODE DID NOT SET
6762 000434  2$:               REPORT  <USC>           ; CONDITION CODE OK, BRANCH FAILED
6763
6764 000444                    SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
     000444  T36S2:
6765                    ;+
6766                    ; NOW CHECK THE ALU ZERO BRANCH WITH DATA TYPE=WORD
6767                    ;-
6768
6769 000446                    INITIALIZE              ; SET ROM NOP
6770 000450                    BLKMIC  TMP124,,10000,1 ; LOAD THE MICRO INSTRUCTION
6771 000464                    LOOP    I,9,16
6772 000476                    FLTONE  TMP130,I        ; GENERATE THE TEST PATTERN
6773 000504                    ERLOOP
6774 000506                    INITIALIZE
6775 000510                    LDIDREG RWDQ,TMP130     ; LOAD THE DATA PATTERN
6776 000516                    FETCH   10000,,NONOP
6777 000526                    CLOCK   4               ; LATCH THE MICRO CONDITION CODES
```

```
                                                
```

```
6833 001110          ERLOOP
6834 001112          INITIALIZE
6835 001114          LDIDREG RWDQ,TMP130     ; LOAD THE DATA PATTERN
6836 001122          FETCH   10000,,NONOP
6837 001132          CLOCK   4               ; LOAD THE MICRO CONDITION CODES
6838 001136  SYNC89: CLOCK   4               ; EXECUTE THE BRANCH
6839 001142          CMPPCSV XTMP126         ; CHECK FOR NO BRANCH
6840 001150          IFERROR 134,DP12D       ; ALU ZERO BRANCH FAILED, DT=LONG
6841 001156          ENDLOOP I               ; CONTINUE
6842
6843          ;+
6844          ; NOW CHECK THAT THE BIT ASCERTS
6845          ;-
6846
6847 001162          LDIDREG RWDQ,TMP126     ; LOAD ZERO
6848 001170          ERLOOP
6849 001172          INITIALIZE
6850 001174          FETCH   10000,,NONOP
6851 001204          CLOCK   4               ; LOAD THE MICRO CONDITION CODES
6852 001210  SYNC8A: CLOCK   4               ; EXECUTE THE BRANCH
6853 001214          CMPPCSV TMP127          ; CHECK THAT BRANCH BIT ASCERTED
6854 001222          IFERROR 135,DP12E       ; ALU ZERO BRANCH BIT DID NOT ASCERT, DT=LONG
6855 001230          SKIP
6856
6857 001234  DP12D:  INITIALIZE
6858 001236          LDIDREG RWDQ,TMP130     ; LOAD THE TEST PATTERN
6859 001244          FETCH   10000,,NONOP    ; START THE SEQUENCER
6860 001254          CLOCK   2               ; PUT DATA ON ALU OUTPUT
6861 001260          TSTVB   T34V5,I         ; CHECK THE ALU BITS THAT GO TO CEH
6862 001266          CHKPNT  1$              ; BRANCH IF OK
6863 001274          REPORT  <DEP>           ; ALU BIT INCORRECT
6864 001304  1$:     CLOCK   2               ; COMPLETE INSTRUCTION
6865 001310          TSTVB   T34V2           ; CHECK "ALU Z(1)" INACTIVE
6866 001316          CHKPNT  2$              ; BRANCH IF OK
6867 001324          REPORT  <CEH,ICL>       ; "ALU=0" FAILED
6868 001334  2$:     REPORT  <USC>
6869
6870 001344  DP12E:  INITIALIZE
6871 001346          LDIDREG RWDQ,TMP126     ; LOAD ALL ZERO'S
6872 001354          FETCH   10000,,NONOP    ; START THE SEQUENCER
6873 001364          CLOCK   2               ; PUT DATA ON ALU OUTPUT
6874 001370          TSTVB   T34V6           ; CHECK BITS THAT GO TO CEH
6875 001376          CHKPNT  1$              ; BRANCH IF OK
6876 001404          REPORT  <DEP>           ; ALU BITS INCORRECT
6877 001414  1$:     CLOCK   2               ; COMPLETE THE INSTRUCTION
6878 001420          TSTVB   T34V2A          ; CHECK "ALU Z(1)" ACTIVE
6879 001426          CHKPNT  2$              ; BRANCH IF OK
6880 001434          REPORT  <CEH,ICL>
6881 001444  2$:     REPORT  <USC>
6882
6883 001454          TESTDAT
          ;-------------------------------------------------------------------------------
6884 001460  TMP123:
6885          ;*1000: RAMX/Q,AMX/RAMX,ALU/A,DT/BYTE,CLK.UBCC,J/1001
6886 001460          .WORD   10001
6887 001462          .WORD   20
6888 001464          .WORD   4000
```

```
6889 001466            .WORD   600
6890 001470            .WORD   120074
6891 001472            .WORD   1
6892            ;*1001: Z?,J/1000
6893 001474            .WORD   10000
6894 001476            .WORD   0
6895 001500            .WORD   4000
6896 001502            .WORD   600
6897 001504            .WORD   474
6898 001506            .WORD   0
6899
6900 001510  TMP124:
6901            ;*1000: RAMX/Q,AMX/RAMX,ALU/A,DT/WORD,CLK.UBCC,J/1001
6902 001510            .WORD   10001
6903 001512            .WORD   20
6904 001514            .WORD   4000
6905 001516            .WORD   600
6906 001520            .WORD   60074
6907 001522            .WORD   1
6908
6909 001524  TMP125:
6910            ;*1000: RAMX/Q,AMX/RAMX,ALU/A,DT/LONG,CLK.UBCC,J/1001
6911 001524            .WORD   10001
6912 001526            .WORD   20
6913 001530            .WORD   4000
6914 001532            .WORD   600
6915 001534            .WORD   20074
6916 001536            .WORD   1
6917
6918 001540  TMP126: .WORD   0,0
6919 001544  TMP127: .WORD   10001
6920 001546  XTMP126:.WORD   10000
6921 001550  TMP130: .WORD   0,0
6922 001554  TMP131: .WORD   SBC
6923
6924 001556  T34V1:  .WORD   1
6925 001560          VBUSG   1,32,1          ; ALU (7:0)=0 L
6926 001562  T34V1A: .WORD   1
6927 001564          VBUSG   1,32,0          ; ALU (7:0)=0 L
6928 001566  T34V2:  .WORD   1
6929 001570          VBUSG   0,31,0          ; ALU Z(1) H
6930 001572  T34V2A: .WORD   1
6931 001574          VBUSG   0,31,1          ; ALU Z(1) H
6932 001576  T34V3:  .WORD   1
6933 001600          VBUSG   1,31,1          ; ALU (15:8)=0 L
6934 001602  T34V3A: .WORD   1
6935 001604          VBUSG   1,31,0          ; ALU (15:8)=0 L
6936 001606  T34V5:  .WORD   0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
6937 001646          .WORD   6
6938 001650          VBUSG   1,25,0          ; ALU BUFF 16 L
6939 001652          VBUSG   1,24,1          ; ALU BUFF 17 L
6940 001654          VBUSG   1,30,0          ; ALU (30:18)=0 L
6941 001656          VBUSG   1,20,1          ; ALU BUFF 31 L
6942 001660          VBUSG   1,31,0          ; ALU (15:8)=0 L
6943 001662          VBUSG   1,32,0          ; ALU (7:0)=0 L
6944 001664          .WORD   6
6945 001666          VBUSG   1,25,1          ; ALU BUFF 16 L
```

```
 6946 001670            VBUSG   1,24,0          ; ALU BUFF 17 L
 6947 001672            VBUSG   1,30,0          ; ALU (30:18)=0 L
 6948 001674            VBUSG   1,20,1          ; ALU BUFF 31 L
 6949 001676            VBUSG   1,31,0          ; ALU (15:8)=0 L
 6950 001700            VBUSG   1,32,0          ; ALU (7:0)=0 L
 6951 001702            .WORD   1
 6952 001704            VBUSG   1,30,1          ; ALU (30:18)=0 L
 6953 001706            .WORD   1
 6954 001710            VBUSG   1,30,1          ; ALU (30:18)=0 L
 6955 001712            .WORD   1
 6956 001714            VBUSG   1,30,1          ; ALU (30:18)=0 L
 6957 001716            .WORD   1
 6958 001720            VBUSG   1,30,1          ; ALU (30:18)=0 L
 6959 001722            .WORD   1
 6960 001724            VBUSG   1,30,1          ; ALU (30:18)=0 L
 6961 001726            .WORD   1
 6962 001730            VBUSG   1,30,1          ; ALU (30:18)=0 L
 6963 001732            .WORD   1
 6964 001734            VBUSG   1,30,1          ; ALU (30:18)=0 L
 6965 001736            .WORD   1
 6966 001740            VBUSG   1,30,1          ; ALU (30:18)=0 L
 6967 001742            .WORD   1
 6968 001744            VBUSG   1,30,1          ; ALU (30:18)=0 L
 6969 001746            .WORD   1
 6970 001750            VBUSG   1,30,1          ; ALU (30:18)=0 L
 6971 001752            .WORD   1
 6972 001754            VBUSG   1,30,1          ; ALU (30:18)=0 L
 6973 001756            .WORD   1
 6974 001760            VBUSG   1,30,1          ; ALU (30:18)=0 L
 6975 001762            .WORD   1
 6976 001764            VBUSG   1,30,1          ; ALU (30:18)=0 L
 6977 001766            .WORD   6
 6978 001770            VBUSG   1,25,1          ; ALU BUFF 16 L
 6979 001772            VBUSG   1,24,1          ; ALU BUFF 17 L
 6980 001774            VBUSG   1,30,0          ; ALU (30:18)=0 L
 6981 001776            VBUSG   1,20,0          ; ALU BUFF 31 L
 6982 002000            VBUSG   1,31,0          ; ALU (15:8)=0 L
 6983 002002            VBUSG   1,32,0          ; ALU (7:0)=0 L
 6984
 6985 002004  T34V6:    .WORD   6
 6986 002006            VBUSG   1,25,1          ; ALU BUFF 16 L
 6987 002010            VBUSG   1,24,1          ; ALU BUFF 17 L
 6988 002012            VBUSG   1,30,0          ; ALU (30:18)=0 L
 6989 002014            VBUSG   1,20,1          ; ALU BUFF 31 L
 6990 002016            VBUSG   1,31,0          ; ALU (15:8)=0 L
 6991 002020            VBUSG   1,32,0          ; ALU (7:0)=0 L
 6992 002022            ENDDAT
                ;-----------------------------------------------------------------------------
 6993
 6994 002022            FORCEOVERLAY
                .SBTTL  SECTION NUMBER 13
      000000            .PSECT  OVR23
 6995
```

```
                .SBTTL  T37     SCRATCH PAD DATA INTEGRITY
                ;;*****************************************************************************
                ;++
                ;TEST   37      SCRATCH PAD DATA INTEGRITY
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST FLOATS A ONE AND A ZERO THRU SCRATCH PADS RA0, RB0,
                ;       AND RC0.
                ;
                ;       SUBTST 1 - FLOAT A ZERO THRU RA[0]
                ;       SUBTST 2 - FLOAT A ONE THRU RA[0]
                ;       SUBTST 3 - FLOAT A ZERO THRU RB[0]
                ;       SUBTST 4 - FLOAT A ONE THRU RB[0]
                ;       SUBTST 5 - FLOAT A ZERO THRU RC[0]
                ;       SUBTST 6 - FLOAT A ONE THRU RC[0]
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE RA, RB, AND RC SCRATCH PADS AND THE AMX
                ;       AND BMX SELECTS (FOR THE SCRATCH PADS) ON THE DAP,DCP,DDP, AND DEP
                ;       MODULES.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED CONTENTS OF THE SCRATCH PAD
                ;             RECEIVED CONTENTS OF THE SCRATCH PAD
                ;             LOOP COUNT -- INDICATES THE POSITION OF THE FLOATING ZERO OR ONE,
                ;               I.E. 1=BIT 0, 2=BIT 1, 3=BIT 2, ETC.
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNC8B--RA[0] IS LOADED
                ;                  SYNC8C--RA[0] IS READ
                ;       SUBTST 2 - SYNC8D--RA[0] IS LOADED
                ;                  SYNC8E--RA[0] IS READ
                ;       SUBTST 3 - SYNC8F--RB[0] IS LOADED
                ;                  SYNC90--RB[0] IS READ
                ;       SUBTST 4 - SYNC91--RB[0] IS LOADED
                ;                  SYNC92--RB[0] IS READ
                ;       SUBTST 5 - SYNC93--RC[0] IS LOADED
                ;                  SYNC94--RC[0] IS READ
                ;       SUBTST 6 - SYNC95--RC[0] IS LOADED
                ;                  SYNC96--RC[0] IS READ
                ;--
                ;;*****************************************************************************
    000034      T37:
7039 000040             INITIALIZE
7040 000042             BLKMIC  RARW,,10000,2   ; LOAD MICROWORDS TO R/W RA0
7041
7042 000056             SUBTEST
                ;//////////////////////////////////////////////////////////////////////////////
    000056      T37S1:
7043            ;+
7044            ; NOW FLOAT A ZERO THRU RA0
7045            ;-
7046
7047 000060             LOOP    I,1,32          ; SET THE LOOP COUNT
7048 000072             ERLOOP
7049 000074             FLTZRO  TMP500,I        ; GENERATE TEST PATTERN
7050 000102             INITIALIZE
```

T37 SCRATCH PAD DATA INTEGRITY

```
7051 000104           LDIDREG RWDQ,TMP500     ; PUT PATTERN INTO Q REGISTER
7052 000112           FETCH   10000,,NONOP    ; START EXECUTION
7053 000122  SYNC8B:  CLOCK   4               ; WRITE RA[0]
7054 000126  SYNC8C:  CLOCK   4               ; READ RA[0]
7055 000132           READID  RWDQ            ; READ THE D REGISTER
7056 000136           CMPCAD  EQ,IDREGLO,TMP500 ; CHECK THE DATA THAT CAME BACK
7057 000162           IFERROR ,GPR1           ; RA SCRATCH PAD FAILED
7058 000170           ENDLOOP I               ; CONTINUE
7059
7060 000174           SUBTEST
                ;!!!
```

```
7102 000446          LOOP    I,1,32          ; SET THE LOOP COUNT
7103 000460          ERLOOP
7104 000462          FLTONE  TMP500,I        ; GENERATE THE TEST PATTERN
7105 000470          INITIALIZE
7106 000472          LDIDREG RWDQ,TMP500     ; LOAD THE TEST PATTERN
7107 000500          FETCH   10000,,NONOP    ; START EXECUTION
7108 000510  SYNC91: CLOCK   4               ; WRITE RB[0]
7109 000514  SYNC92: CLOCK   4               ; READ RB[0]
7110 000520          READID  RWDQ            ; GET THE DATA BACK
7111 000524          CMPCAD  EQ,IDREGLO,TMP500 ; CHECK THE RESULT
7112 000550          IFERROR ,GPR1A          ; RB0 FAILED
7113 000556          ENDLOOP I               ; CONTINUE
7114
7115 000562          SUBTEST
             ;////////////////////////////////////////////////////////////////////////////////
     000562  T37S5:
7116         ;+
7117         ; NOW CHECK RC0
7118         ;-
7119
7120 000564          INITIALIZE
7121 000566          BLKMIC  RCRW,,10000,2   ; LOAD MICROWORDS TO W/R RC0
7122 000602          LOOP    I,1,32          ; SET THE LOOP COUNT
7123 000614          ERLOOP
7124 000616          FLTZRO  TMP500,I        ; GENERATE TEST PATTERN
7125 000624          INITIALIZE
7126 000626          LDIDREG RWDQ,TMP500     ; PUT PATTERN INTO Q REGISTER
7127 000634          FETCH   10000,,NONOP    ; START EXECUTION
7128 000644  SYNC93: CLOCK   4               ; WRITE RC[0]
7129 000650  SYNC94: CLOCK   4               ; READ RC[0]
7130 000654          READID  RWDQ            ; READ THE D REGISTER
7131 000660          CMPCAD  EQ,IDREGLO,TMP500 ; CHECK THE DATA THAT CAME BACK
7132 000704          IFERROR ,GPR1           ; RC SCRATCH PAD FAILED
7133 000712          ENDLOOP I               ; CONTINUE
7134
7135 000716          SUBTEST
             ;////////////////////////////////////////////////////////////////////////////////
     000716  T37S6:
7136         ;+
7137         ; NOW FLOAT A ONE THRU THE REGISTER
7138         ;-
7139
7140 000720          LOOP    I,1,32          ; SET THE LOOP COUNT
7141 000732          ERLOOP
7142 000734          FLTONE  TMP500,I        ; GENERATE THE TEST PATTERN
7143 000742          INITIALIZE
7144 000744          LDIDREG RWDQ,TMP500     ; LOAD THE TEST PATTERN
7145 000752          FETCH   10000,,NONOP    ; START EXECUTION
7146 000762  SYNC95: CLOCK   4               ; WRITE RC[0]
7147 000766  SYNC96: CLOCK   4               ; READ RC[0]
7148 000772          READID  RWDQ            ; GET THE DATA BACK
7149 000776          CMPCAD  EQ,IDREGLO,TMP500 ; CHECK THE RESULT
7150 001022          IFERROR ,GPR1A          ; RC0 FAILED
7151 001030          ENDLOOP I               ; CONTINUE
7152 001034          SKIP
7153
7154 001040  GPR1:  MASK    TMP500,TBYTE0   ; GET EXPECTED BYTE 0
```

```
7155 001046          CMPCAM  ,IDREGLO,TBYTE0,,TMP500 ; IS BYTE 0 BAD?
7156 001062          CHKPNT  1$              ; BRANCH IF NO
7157 001070          FLTZRO  TMP500,I        ; REGENERATE TEST PATTERN
7158 001076          MASK    TMP500,TBYTE1   ; GET EXPECTED BYTE 1
7159 001104          CMPCAM  ,IDREGLO,TBYTE1,,TMP500 ; IS BYTE 1 BAD?
7160 001120          CHKPNT  2$              ; BRANCH IF NO
7161 001126          CMPCA   ,IDREGHI,TMP500+2 ; BYTES 2,3 BAD?
7162 001140          CHKPNT  3$              ; BRANCH IF NO
7163 001146          REPORT  <DAP>           ; ALL BYTES BAD (CHECK WRITE ENABLE)
7164 001156  2$:     REPORT  <DCP>           ; BYTE 0 BAD BUT NOT BYTE 1
7165 001166  3$:     REPORT  <DCP,DDP>       ; BYTES 0,1 BAD, BYTES 2,3 GOOD
7166
7167 001176  1$:     FLTZRO  TMP500,I        ; REGENERATE TEST PATTERN
7168 001204          MASK    TMP500,TBYTE1   ; GET EXPECTED BYTE 1
7169 001212          CMPCAM  ,IDREGLO,TBYTE1,,TMP500 ; BYTE 1 BAD?
7170 001226          CHKPNT  4$              ; BRANCH IF NO
7171 001234          REPORT  <DDP>           ; BYTE 0 GOOD, BYTE 1 BAD
7172 001244  4$:     CMPCA   ,IDREGHI,TMP500+2 ; BYTES 2,3 BAD?
7173 001256          CHKPNT  5$              ; BRANCH IF NO
7174 001264          REPORT  <DEP>           ; BYTES 0,1 GOOD, BYTES 2,3 BAD
7175 001274  5$:     REPORT  <DAP,DCP,DDP,DEP> ; ALL BYTES GOOD (SHOULD NEVER HAPPEN)
7176
7177
7178 001304  GPR1A:  MASK    TMP500,TBYTE0   ; GET EXPECTED BYTE 0
7179 001312          CMPCAM  ,IDREGLO,TBYTE0,,TMP500 ; BYTE 0 GOOD?
7180 001326          CHKPNT  1$              ; BRANCH IF YES
7181 001334          FLTONE  TMP500,I        ; REGENERATE TEST PATTERN
7182 001342          MASK    TMP500,TBYTE1   ; GET EXPECTED BYTE 1
7183 001350          CMPCAM  ,IDREGLO,TBYTE1,,TMP500 ; BYTE 1 BAD?
7184 001364          CHKPNT  2$              ; BRANCH IF NO
7185 001372          CMPCA   ,IDREGHI,TMP500+2 ; BYTES 2,3 BAD?
7186 001404          CHKPNT  3$              ; BRANCH IF NO
7187 001412          REPORT  <DAP>           ; ALL BYTES BAD
7188 001422  2$:     REPORT  <DCP>           ; BYTE 0 BAD, BYTE 1 GOOD
7189 001432  3$:     REPORT  <DCP,DDP>       ; BYTES 0,1 BAD, BYTES 2,3 GOOD
7190
7191 001442  1$:     FLTONE  TMP500,I        ; REGENERATE TEST PATTERN
7192 001450          MASK    TMP500,TBYTE1   ; GET EXPECTED BYTE1
7193 001456          CMPCAM  ,IDREGLO,TBYTE1,,TMP500 ; BYTE 1 BAD?
7194 001472          CHKPNT  4$              ; BRANCH IF NO
7195 001500          REPORT  <DDP>           ; BYTE 0 GOOD, BYTE 1 BAD
7196 001510  4$:     CMPCA   ,IDREGHI,TMP500+2 ; BYTES 2,3 BAD?
7197 001522          CHKPNT  5$              ; BRANCH IF NO
7198 001530          REPORT  <DEP>           ; BYTES 0,1 GOOD, BYTES 2,3 BAD
7199 001540  5$:     REPORT  <DAP,DCP,DDP,DEP> ; ALL BYTES GOOD (SHOULD NEVER HAPPEN)
7200
7201 001550          TESTDAT
                     ;-------------------------------------------------------------------------------
7202 001554  TMP500: .WORD   0,0
7203 001560  TMP501: .WORD   SBC
7204
7205                 ;*1000: R[0] Q,J/1001
7206 001562  RARW:   .WORD   10001
7207 001564          .WORD   0
7208 001566          .WORD   5200
7209 001570          .WORD   600
7210 001572          .WORD   20074
```

```
                        .WORD   1

                ;*1001: D_R[0],J/1001
001576                  .WORD   10001
001600                  .WORD   0
001602                  .WORD   5000
001604                  .WORD   600
001606                  .WORD   74
001610                  .WORD   4000

                ;*1000: R[0]_Q,J/1001
001612  RBRW:          .WORD   10001
001614                  .WORD   0
001616                  .WORD   5200
001620                  .WORD   600
001622                  .WORD   20074
001624                  .WORD   1

                ;*1001: D_RB[0],J/1001
001626                  .WORD   10001
001630                  .WORD   0
001632                  .WORD   5000
001634                  .WORD   600
001636                  .WORD   70
001640                  .WORD   4014

                ;*1000: RC[0]_Q,J/1001
001642  RCRW:          .WORD   10001
001644                  .WORD   0
001646                  .WORD   4600
001650                  .WORD   600
001652                  .WORD   20074
001654                  .WORD   1

                ;*1001: D_RC[0],J/1001
001656                  .WORD   10001
001660                  .WORD   0
001662                  .WORD   4400
001664                  .WORD   600
001666                  .WORD   70
001670                  .WORD   4020

001672  TBYTE0: .WORD   377
001674  TBYTE1: .WORD   177400
001676          ENDDAT
                ;-----------------------------------------------------------------------------


                        FORCEOVERLAY
                .SBTTL  SECTION NUMBER 14
000000                  .PSECT  OVR24
```

```
                .SBTTL  T38     SCRATCH PAD DUAL ADDRESSING
                ;;*****************************************************************************
                ;++
                ;TEST   38      SCRATCH PAD DUAL ADDRESSING
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THAT NONE OF THE SCRATCH PAD ADDRESS BITS ARE
                ;       STUCK AT ZERO OR ONE. THIS IS DONE BY WRITING THE ADDRESS OF
                ;       THE SCRATCH PAD IN ITS LOCATION (EACH 4 BITS GET THE ADDRESS) AND
                ;       THEN REAING THEM BACK. THE ERROR LOOP FOR THESE TESTS INCLUDES
                ;       WRITING ALL THE SCRATCH PADS.
                ;
                ;       SUBTEST 1 - CHECK THE RA SCRATCH PADS
                ;       SUBTEST 2 - CHECK THE RB SCRATCH PADS
                ;       SUBTEST 3 - CHECK THE RC SCRATCH PADS
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE SCRATCH PAD ADDRESS MUX'S ON THE DAP MODULE AND
                ;       THE SCRATCH PAD CHIPS ON THE DCP, DDP, AND DEP MODULES.
                ;
                ; ERROR DESCRIPTION
                ;       DATA:   EXPECTED SCRATCH PAD DATA
                ;               RECIEVED SCRATCH PAD DATA
                ;
                ;       NOTE:   THE EXPECTED AND RECIEVED SCRATCH PAD ADDRESS IS CONTAINED
                ;               IN EACH DIGIT OF THE DATA.
                ;--
                ;;*****************************************************************************
    000034      T38:
7291
7292 000040             INITIALIZE
7293 000042             BLKMIC  T3AL4,,10001,1  ; LOAD A NOP INTO 1001(H)
7294 000056             BLKMIC  T3AL4,,10400,1  ; PUT NOP IN INIT VECTOR
7295 000072             BLKMIC  T3AL4,,10417,1  ; PUT NOP IN PAR ERR VECTOR
7296 000106             LOADCA  CONMCR,T3AL5    ; SET THE FORCE UPC BIT 12 BIT
7297
7298 000116             SUBTEST
                ;////////////////////////////////////////////////////////////////////////////
    000116      T38S1:
7299            ;+
7300            ; LOAD THE RA SCRATCH PADS WITH THE PATTERN
7301            ;-
7302
7303 000120             ERLOOP                  ; SETUP AN ERROR LOOP
7304 000122             LOOP    I,1,16          ; SET THE LOOP COUNT FOR 16 REGISTERS
7305 000134             SPAGEN  RAWR,I          ; GENERATE THE MICRO WORD TO WRITE
7306 000142             BLKMIC  RAWR,,10000,1   ; LOAD THE MICRO WORDS
7307 000156             INITIALIZE
7308 000160             LDIDREG RWDQ,T3AL1,I    ; LOAD THE DATA PATTERN INTO THE D REG
7309 000166             FETCH   10000,,NONOP    ; START EXECUTION
7310 000176             CLOCK   4               ; LOAD THE SCRATCH PAD
7311 000202             ENDLOOP I               ; CONTNUE FOR ALL 16 REGISTERS
7312
7313            ;+
7314            ; NOW READ THE REGISTERS BACK AND CHECK THEM
7315            ;-
7316
```

```
7317 000206          LOOP    I,1,16           ; SET THE LOOP COUNT
7318 000220          SPAGEN  RARE,I           ; GENERATE THE MICRO WORD
7319 000226          BLKMIC  RARE,,10000,1    ; LOAD THE MICRO WORD
7320 000242          INITIALIZE
7321 000244          FETCH   10000,,NONOP     ; START EXECUTION
7322 000254          CLOCK   4                ; READ THE REGISTER INTO THE D REG
7323 000260          READID  RWDQ             ; READ THE D REGISTER
7324 000264          CMPCAD  ,IDREGLO,T3AL1,I ; CHECK RECIEVED DATA
7325 000310          IFERROR ,T3AF1           ; IF ERROR GO TO 'T3AF1'
7326 000316          ENDLOOP I                ; CHECK ALL 16 REGISTERS
7327
7328
7329 000322          SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
     000322     T38S2:
7330                ;+
7331                ; NOW CHECK THE RB SCRATCH PAD DUAL ADDRESSING
7332                ;
7333                ; FIRST WRITE THE DATA PATTERNS TO ALL THE REGISTERS
7334                ;-
7335
7336 000324          ERLOOP                   ; SETUP AN ERROR LOOP
7337 000326          LOOP    I,1,16           ; SET THE LOOP COUNT FOR 16 REGISTERS
7338 000340          SPAGEN  RAWR,I           ; GENERATE THE MICRO WORD TO WRITE
7339 000346          BLKMIC  RAWR,,10000,1    ; LOAD THE MICRO WORDS
7340 000362          INITIALIZE
7341 000364          LDIDREG RWDQ,T3AL1,I     ; LOAD THE DATA PATTERN INTO THE D REG
7342 000372          FETCH   10000,,NONOP     ; START EXECUTION
7343 000402          CLOCK   4                ; LOAD THE SCRATCH PAD
7344 000406          ENDLOOP I                ; CONTNUE FOR ALL 16 REGISTERS
7345
7346                ;+
7347                ; NOW READ THE REGISTERS BACK AND CHECK THEM
7348                ;-
7349
7350 000412          LOOP    I,1,16           ; SET THE LOOP COUNT
7351 000424          SPAGEN  RBRE,I           ; GENERATE THE MICRO WORD
7352 000432          BLKMIC  RBRE,,10000,1    ; LOAD THE MICRO WORD
7353 000446          INITIALIZE
7354 000450          FETCH   10000,,NONOP     ; START EXECUTION
7355 000460          CLOCK   4                ; READ THE REGISTER INTO THE D REG
7356 000464          READID  RWDQ             ; READ THE D REGISTER
7357 000470          CMPCAD  ,IDREGLO,T3AL1,I ; CHECK RECIEVED DATA
7358 000514          IFERROR ,T3AF1           ; IF ERROR GO TO 'T3AF1'
7359 000522          ENDLOOP I                ; CHECK ALL 16 REGISTERS
7360
7361                ;+
7362                ; CLEAR THE RB AND RA SCRATCH PADS
7363                ;-
7364
7365 000526          LOOP    I,1,16
7366 000540          SPAGEN  RAWR,I           ; GENERATE MICROWORD
7367 000546          BLKMIC  RAWR,,10000,1    ; LOAD THE INSTRUCTION
7368 000562          INITIALIZE
7369 000564          LDIDREG RWDQ,T3AL1       ; LOAD DATA
7370 000572          FETCH   10000,,NONOP     ; START EXECUTION
7371 000602          CLOCK   4                ; CLEAR THE REGISTER
```

```
                        ENDLOOP I                   ; CONTINUE


                        SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
        000612  T38S3:
                ;+
                ; NOW CHECK THE RC SCRATCH PADS
                ;-

                        ERLOOP                      ; SETUP AN ERROR LOOP
                        LOOP    I,1,16              ; SET THE LOOP COUNT FOR 16 REGISTERS
                        SPAGEN  RCWR,I              ; GENERATE THE MICRO WORD TO WRITE
                        BLKMIC  RCWR,,10000,1       ; LOAD THE MICRO WORDS
                        INITIALIZE
                        LDIDREG RWDQ,T3AL1,I        ; LOAD THE DATA PATTERN INTO THE D REG
                        FETCH   10000,,NONOP        ; START EXECUTION
                        CLOCK   4                   ; LOAD THE SCRATCH PAD
                        ENDLOOP I                   ; CONTNUE FOR ALL 16 REGISTERS

                ;+
                ; NOW READ THE REGISTERS BACK AND CHECK THEM
                ;-

                        LOOP    I,1,16              ; SET THE LOOP COUNT
                        SPAGEN  RCRE,I              ; GENERATE THE MICRO WORD
                        BLKMIC  RCRE,,10000,1       ; LOAD THE MICRO WORD
                        INITIALIZE
                        FETCH   10000,,NONOP        ; START EXECUTION
                        CLOCK   4                   ; READ THE REGISTER INTO THE D REG
                        READID  RWDQ                ; READ THE D REGISTER
                        CMPCAD  ,IDREGLO,T3AL1,I    ; CHECK RECIEVED DATA
                        IFERROR ,T3AF1              ; IF ERROR GO TO 'T3AF1'
                        ENDLOOP I                   ; CHECK ALL 16 REGISTERS
                        LOADCA  CONMCR,T3AL6        ; CLEAR THE FORCE UPC BIT 12 BIT
                        SKIP


        001032  T3AF1:  MOVE    T3AL1,I,T3AL2,LONG  ; MOVE THE INPUT DATA TO A WORKING AREA
                        MOVE    T3AL1+2,I,T3AL2+2,LONG ; ...
                        MASK    T3AL2,T3AL3         ; THROW AWAY ALL BUT BYTE 0
                        CMPCAM  ,IDREGLO,T3AL3,,T3AL2 ; IS BYTE 0 OK?
                        CHKPNT  1$                  ; BRANCH IF YES
                        MOVE    T3AL1,I,T3AL2,LONG  ; GET TEST PATTERN BACK AGAIN
                        MASK    T3AL2,T3AL3+2       ; THROW AWAY ALL BUT BYTE 1
                        CMPCAM  ,IDREGLO,T3AL3+2,,T3AL2 ; IS BYTE 1 OK?
                        CHKPNT  2$                  ; BRANCH IF YES
                        CMPCA   ,IDREGHI,T3AL2+2    ; IS UPPER 16 BITS OK?
                        CHKPNT  3$                  ; BRANCH IF YES
                        REPORT  <DAP>               ; ALL BYTES ARE BAD
        001172  2$:     REPORT  <DCP>               ; BYTE 0 BAD, BUT BYTE 1 IS GOOD
        001202  3$:     REPORT  <DCP,DDP>           ; BYTES 0 AND 1 BAD, BUT BYTES 2,3 GOOD

                ;+
                ; BYTE 0 IS GOOD, CHECK BYTES 1,2, AND 3
                ;-
```

```
7427 001212  1$:     MOVE    T3AL1,I,T3AL2,LONG ; GET TEST PATTERN
7428 001222          MASK    T3AL2,T3AL3+2    ; THROW AWAY BYTE 0
7429 001230          CMPCAM  ,IDREGLO,T3AL3+2,,T3AL2 ; IS BYTE 1 OK?
7430 001244          CHKPNT  4$               ; BRANCH IF YES
7431 001252          REPORT  <DDP>            ; BYTE 0 OK, BUT BYTE 1 BAD
7432 001262  4$:     CMPCA   ,IDREGHI,T3AL2+2 ; ARE BYTES 2,3 OK?
7433 001274          CHKPNT  5$               ; BRANCH IF YES
7434 001302          REPORT  <DEP>            ; BYTES 0,1 GOOD, BUT BYTES 2,3 ARE BAD
7435 001312  5$:     REPORT  <DAP,DCP,DDP,DEP> ; ALL BYTES ARE GOOD??????
7436
7437
7438 001322          TESTDAT
                ;-------------------------------------------------------------------------
7439            ;+
7440            ; FOLLOWING ARE THE SIXTEEN DATA PATTERNS USED FOR THIS TEST
7441            ;-
7442
7443 001326  T3AL1:  .WORD   0,0              ; HEX 0,0
7444 001332          .WORD   10421,10421      ; HEX 1111,1111
7445 001336          .WORD   21042,21042      ; HEX 2222,2222
7446 001342          .WORD   31463,31463      ; HEX 3333,3333
7447 001346          .WORD   42104,42104      ; HEX 4444,4444
7448 001352          .WORD   52525,52525      ; HEX 5555,5555
7449 001356          .WORD   63146,63146      ; HEX 6666,6666
7450 001362          .WORD   73567,73567      ; HEX 7777,7777
7451 001366          .WORD   104210,104210    ; HEX 8888,8888
7452 001372          .WORD   114631,114631    ; HEX 9999,9999
7453 001376          .WORD   125252,125252    ; HEX AAAA,AAAA
7454 001402          .WORD   135673,135673    ; HEX BBBB,BBBB
7455 001406          .WORD   146314,146314    ; HEX CCCC,CCCC
7456 001412          .WORD   156735,156735    ; HEX DDDD,DDDD
7457 001416          .WORD   167356,167356    ; HEX EEEE,EEEE
7458 001422          .WORD   177777,177777    ; HEX FFFF,FFFF
7459
7460            ;+
7461            ; FOLLOWING ARE THE MICRO WORDS TO LOAD THE SCRATCH PADS
7462            ;-
7463
7464 001426  RAWR:
7465 001426  RBWR:
7466            ;*1000: R[]_Q,J/1001
7467
7468 001426          .WORD   10001
7469 001430          .WORD   0
7470 001432          .WORD   5200
7471 001434          .WORD   600
7472 001436          .WORD   20074
7473 001440          .WORD   1
7474
7475 001442  RCWR:
7476            ;*1000: RC[]_Q,J/1001
7477
7478 001442          .WORD   10001
7479 001444          .WORD   0
7480 001446          .WORD   4600
7481 001450          .WORD   600
7482 001452          .WORD   20074
```

```
   7483 001454            .WORD   1
   7484
   7485            ;+
   7486            ; THE FOLLOWING MICRO INSTRUCTIONS ARE USED TO READ THE SCRATCH PADS.
   7487            ;-
   7488
   7489 001456  RARE:
   7490            ;*1000: D_RA[],J/1001
   7491
   7492 001456            .WORD   10001
   7493 001460            .WORD   0
   7494 001462            .WORD   5000
   7495 001464            .WORD   600
   7496 001466            .WORD   74
   7497 001470            .WORD   4000
   7498
   7499 001472  RBRE:
   7500            ;*1000: D_RC[],J/1001
   7501
   7502 001472            .WORD   10001
   7503 001474            .WORD   0
   7504 001476            .WORD   5000
   7505 001500            .WORD   600
   7506 001502            .WORD   70
   7507 001504            .WORD   4014
   7508
   7509 001506  RCRE:
   7510            ;*1000: D_RC[],J/1001
   7511
   7512 001506            .WORD   10001
   7513 001510            .WORD   0
   7514 001512            .WORD   4400
   7515 001514            .WORD   600
   7516 001516            .WORD   70
   7517 001520            .WORD   4020
   7518
   7519 001522  T3AL2:    .WORD   0,0                     ; WORKING LOCATIONS FOR BIT SLICE TESTS
   7520 001526  T3AL3:    .WORD   377,177400              ; BYTE MASKS FOR BIT SLICE TESTS
   7521 001532  T3AL4:
   7522            ;*1001: NOP,J/1001
   7523
   7524 001532            .WORD   10001
   7525 001534            .WORD   0
   7526 001536            .WORD   4000
   7527 001540            .WORD   600
   7528 001542            .WORD   74
   7529 001544            .WORD   0
   7530
   7531 001546  T3AL5:    .WORD   UPC12+SBC
   7532 001550  T3AL6:    .WORD   SBC
   7533 001552            ENDDAT
                ;-------------------------------------------------------------------------------
   7534
   7535
   7536 001552            FORCEOVERLAY
                .SBTTL  SECTION NUMBER 15
        000000            .PSECT  OVR25
```

7537

```
7559                .SBTTL  T39     ID MAINT LOCKOUT WHILE CLK RUNNING
                    ;;*****************************************************************************
                    ;++
                    ;TEST   39      ID MAINT LOCKOUT WHILE CLK RUNNING
                    ;
                    ; TEST DESCRIPTION
                    ;       THIS TEST CHECKS THAT THE ID MAINTENANCE BIT DOES NOT CAUSE AN ID
                    ;       BUS CYCLE WHEN THE CLOCK IS RUNNING.
                    ;
                    ; LOGIC DESCRIPTION
                    ;       THIS TEST CHECKS THE PRESET LOGIC ON THE ID MAINTENANCE BIT ON
                    ;       THE CIB MODULE.
                    ;
                    ; ERROR DESCRIPTION
                    ;       DATA: NOT EXPECTED CONTENTS OF THE FROM ID REG
                    ;             RECEIVED CONTENTS OF THE FROM ID REG
                    ;
                    ;       NOTE: IF THIS TEST FAILS IT MEANS THAT THE FROM ID REG WAS WRITTEN WITH
                    ;             THE DATA IN THE TO ID REG WHEN THE ID MAINTENANCE BIT WAS SET.
                    ;
                    ; SYNC POINT DESCRIPTION
                    ;       ADDRESS SYNC9F--ID MAINT IS WRITTEN TO
                    ;--
                    ;;*****************************************************************************
    000034          T39:
7563 000040                 INITIALIZE
7564 000042                 BLKMIC  T33L1,,10000,1  ; LOAD A NOP MICRO INSTRUCTION
7565 000056                 ERLOOP
7566 000060                 INITIALIZE
7567 000062                 FETCH   10000,,NONOP    ; EXECUTE THE MICRO INSTRUCTION
7568 000072                 LOADCA  CONMCR,T33L2    ; START THE CLOCK
7569 000102                 LOADCA  TOIDLO,T33L3    ; PUT DATA IN TO ID LO
7570 000112                 LOADCA  TOIDHI,T33L3+2  ; AND THE HI 16 BITS
7571 000122     SYNC9F:     LOADCA  IDCS,T33L4      ; TRY AND SET THE MAINTENANCE BIT
7572 000132                 LOADCA  CONMCR,T33L5    ; STOP THE CLOCK
7573 000142                 CMPCAD  NE,FMIDLO,T33L3 ; CHECK THAT FM ID REG WAS NOT WRITTEN
7574 000166                 IFERROR ,CIB33          ;
7575 000174                 SKIP
7576
7577 000200     CIB33:      REPORT  <CIB>
7578
7579 000210                 TESTDAT
                    ;-----------------------------------------------------------------------------
7580 000214     T33L1:
7581                ;*1000: BEN/0,J/1000
7582 000214                 .WORD   10000
7583 000216                 .WORD   0
7584 000220                 .WORD   4000
7585 000222                 .WORD   600
7586 000224                 .WORD   74
7587 000226                 .WORD   0
7588
7589 000230     T33L2:      .WORD   PROCEED
7590 000232     T33L3:      .WORD   125252,125252
7591 000236     T33L4:      .WORD   IDMAINT+IDWRITE+CONTXD
7592 000240     T33L5:      .WORD   SBC
7593 000242                 ENDDAT
```

T39   ID MAINT LOCKOUT WHILE CLK RUNNING

```
                ;-------------------------------------------------------------------------------
   7594
```

```
                        .SBTTL  T3A     RUN BIT IN THE MCS REGISTER
                        ;;*****************************************************************************
                        ;++
                        ;TEST   3A      RUN BIT IN THE MCS REGISTER
                        ;
                        ; TEST DESCRIPTION
                        ;       THIS TEST CHECKS THAT AN INTERRUPT STROBE FUNCTION BY THE MICRO CODE
                        ;       CAUSES THE RUN BIT IN THE MCR REGISTER TO SET.
                        ;
                        ; LOGIC DESCRIPTION
                        ;       THIS TEST CHECKS THE LOGIC ON THE ICL MODULE THAT DECODES THE INTERRUPT
                        ;       STROBE FUNCTION AND THE ONE-SHOT ON THE CIB MODULE THAT DRIVES THE
                        ;       RUN BIT IN THE MCR REGISTER.
                        ;
                        ; ERROR DESCRIPTION
                        ;       DATA: EXPECTED CONTENTS OF THE MCR REG
                        ;             RECEIVED CONTENTS OF THE MCR REG
                        ;
                        ; SYNC POINT DESCRIPTION
                        ;       ADDRESS SYNCA7--RUN ONE-SHOT SHOULD BE SETTING EVERY CYCLE
                        ;--
                        ;;*****************************************************************************
            000242      T3A:
 7618 000246                    INITIALIZE
 7619
 7620 000250                    BLKMIC  T34L1,,10000,2  ; LOAD MICRO INSTRUCTION TO SET INTRPT STROBE
 7621 000264                    ERLOOP
 7622 000266                    INITIALIZE
 7623 000270                    FETCH   10000,,NONOP    ; GET THE MICRO INSTRUCTION
 7624 000300                    LOADCA  CONMCR,T34L2    ; START THE CLOCK
 7625 000310    SYNCA7: CMPCAM  EQ,CONMCS,T34L3,,T34L3 ; CHECK THAT RUN BIT IS SET
 7626 000324                    LOADCA  CONMCR,T34L4    ; STOP THE CLOCK
 7627 000334                    IFERROR ,CIB34          ; RUN BIT IS NOT SET
 7628 000342                    SKIP
 7629
 7630 000346    CIB34:  REPORT  <ICL,CIB>
 7631
 7632 000356                    TESTDAT
                        ;-----------------------------------------------------------------------------
 7633 000362    T34L1:
 7634                   ;*1000: INTRPT.STROBE,J/1001
 7635 000362                    .WORD   10001
 7636 000364                    .WORD   40000
 7637 000366                    .WORD   4000
 7638 000370                    .WORD   600
 7639 000372                    .WORD   74
 7640 000374                    .WORD   0
 7641
 7642                   ;*1001: BEN/0,J/1000
 7643 000376                    .WORD   10000
 7644 000400                    .WORD   0
 7645 000402                    .WORD   4000
 7646 000404                    .WORD   600
 7647 000406                    .WORD   74
 7648 000410                    .WORD   0
 7649
 7650 000412    T34L2:  .WORD   PROCEED
```

T3A RUN BIT IN THE MCS REGISTER

```
   7651 000414  T34L3:   .WORD    CPURUN
   7652 000416  T34L4:   .WORD    CLRUWRD+SBC
   7653 000420  T34L5:   .WORD    SBC
   7654 000422           ENDDAT
                ;-------------------------------------------------------------------------------
   7655
```

```
                        .SBTTL  T3B     CONSOLE ACK BIT IN THE MCS REGISTER
                        ;;*****************************************************************************
                        ;++
                        ;TEST   3B      CONSOLE ACK BIT IN THE MCS REGISTER
                        ;
                        ; TEST DESCRIPTION
                        ;       THIS TEST CHECKS THAT THE CONSOLE ACKNOWLEDGE BIT IN THE MCS REGISTER
                        ;       SETS WHEN THE MICRO CODE IS EXECUTING A CONSOLE ACK FUNCTION.
                        ;
                        ; LOGIC DESCRIPTION
                        ;       THIS TEST CHECKS THE LOGIC ON THE ICL MODULE THAT DECODES THE CONSOLE
                        ;       ACK FUNCTION AND THE LOGIC ON THE CIB MODULE THAT GATES THIS SIGNAL
                        ;       ONTO THE Q BUS WHEN THE MCS REGISTER IS READ.
                        ;
                        ; ERROR DESCRIPTION
                        ;       DATA: EXPECTED CONTENTS OF THE MCS REGISTER
                        ;             RECEIVED CONTENTS OF THE MCS REGISTER
                        ;
                        ; SYNC POINT DESCRIPTION
                        ;       ADDRESS SYNCA8--CONSOLE ACK SHOULD BE ASSERTED
                        ;--
                        ;;*****************************************************************************
    000422              T3B:
7679 000426                     INITIALIZE
7680
7681 000430                     BLKMIC  T35L1,,10000,1  ; LOAD THE MICRO INSTRUCTION
7682 000444                     ERLOOP
7683 000446                     INITIALIZE
7684 000450                     FETCH   10000,,NONOP    ; READ THE INSTRUCTION
7685 000460                     LOADCA  CONMCR,T35L2    ; START THE CLOCK
7686 000470  SYNCA8:            CMPCAM  EQ,CONMCS,T35L3,,T35L3 ; CHECK THAT THE ACK BIT IS SET
7687 000504                     LOADCA  CONMCR,T35L4    ; STOP THE CLOCK
7688 000514                     IFERROR ,CIB35          ; CONSOLE ACK BIT IS NOT SET
7689 000522                     SKIP
7690
7691 000526  CIB35:             REPORT  <ICL,CIB>
7692
7693 000536                     TESTDAT
                        ;-------------------------------------------------------------------------------
7694 000542  T35L1:
7695                    ;*1000: CID/ACK,J/1000
7696 000542                     .WORD   10000
7697 000544                     .WORD   0
7698 000546                     .WORD   12000
7699 000550                     .WORD   600
7700 000552                     .WORD   74
7701 000554                     .WORD   0
7702
7703 000556  T35L2:            .WORD   PROCEED
7704 000560  T35L3:            .WORD   CONACK
7705 000562  T35L4:            .WORD   CLRUWRD+SBC
7706 000564                     ENDDAT
                        ;-------------------------------------------------------------------------------
7707
```

```
                .SBTTL  T3C     CONSOLE COMMAND MODE BIT IN THE MCS REG
                ;;*****************************************************************************
                ;++
                ;TEST   3C      CONSOLE COMMAND MODE BIT IN THE MCS REG
                ;
                ; TEST DESCRIPTION
                ;       THIS TEST CHECKS THAT THE CONSOLE COMMAND MODE BIT IN THE MCS REGISTER
                ;       SETS AND CLEARS WHEN THE MICRO CODE EXECUTES THE CONSOLE ACK AND
                ;       CONSOLE CONTINUE FUNCTIONS.
                ;
                ;       SUBTST 1 - CHECK THAT THE CM BIT SETS WHEN A CONSOLE ACK IS EXECUTED
                ;       SUBTST 2 - CHECK THAT THE CM BIT CLEARS WHEN A CONSOLE CONT IS EXECUTED.
                ;
                ; LOGIC DESCRIPTION
                ;       THIS TEST CHECKS THE CONSOLE COMMAND MODE FLIP-FLOP ON THE ICL MODULE
                ;       AND THE LOGIC ON THE CIB MODULE THAT GATES THE SIGNAL TO THE Q BUS
                ;       WHEN THE MCS REGISTER IS READ.
                ;
                ; ERROR DESCRIPTION
                ;       DATA: EXPECTED CONTENTS OF THE MCS REGISTER
                ;             RECEIVED CONTENTS OF THE MCS REGISTER
                ;
                ; SYNC POINT DESCRIPTION
                ;       SUBTST 1 - SYNCA9--CM BIT ON ICL SHOULD BE SETTING
                ;       SUBTST 2 - SYNCAA--CM BIT ON ICL SHOULD BE CLEARING
                ;--
                ;;*****************************************************************************
    000564      T3C:
7736 000570             INITIALIZE
7737
7738 000572             SUBTEST
                ;//////////////////////////////////////////////////////////////////////////////
    000572      T3CS1:
7739            ;+
7740            ; CHECK THAT THE BIT SETS
7741            ;-
7742
7743 000574             BLKMIC  T37L1,,10000,2  ; LOAD THE MICRO WORDS
7744 000610             ERLOOP
7745 000612             INITIALIZE
7746 000614             FETCH   10000,,NONOP    ; READUP THE CONSOLE ACK INSTRUCTION
7747 000624     SYNCA9: CLOCK   4               ; EXECUTE THE INSTRUCTION
7748 000630             CMPCAM  EQ,CONMCS,T37L2,,T37L2 ; CHECK THAT THE BIT SET
7749 000644             IFERROR ,CIB37          ; CONSOLE CM BIT DID NOT SET
7750
7751 000652             SUBTEST
                ;//////////////////////////////////////////////////////////////////////////////
    000652      T3CS2:
7752            ;+
7753            ; NOW CHECK THAT THE CM BIT CLEARS
7754            ;-
7755
7756 000654             ERLOOP
7757 000656             INITIALIZE
7758 000660             FETCH   10000,,NONOP    ; FETCH THE CONSOLE ACK
7759 000670             CLOCK   4               ; SET THE BIT
7760 000674     SYNCAA: CLOCK   4               ; CLEAR THE CONSOLE CM BIT
```

T3C CONSOLE COMMAND MODE BIT IN THE MCS REG

```
7761 000700          CMPCAM  EQ,CONMCS,T37L2,,T37L2+2 ; CHECK THAT IT CLEARED
7762 000714          IFERROR ,CIB37           ; CONSOLE CM BIT DID NOT CLEAR
7763 000722          SKIP
7764
7765 000726  CIB37:  REPORT  <ICL,CIB>
7766
7767 000736          TESTDAT
                    ;-------------------------------------------------------------------------------
7768 000742  T37L1:
7769                ;*1000: CID/ACK,J/1001
7770 000742          .WORD   10001
7771 000744          .WORD   0
7772 000746          .WORD   12000
7773 000750          .WORD   600
7774 000752          .WORD   74
7775 000754          .WORD   0
7776
7777                ;*1001: CID/CONT,J/1001
7778 000756          .WORD   10001
7779 000760          .WORD   0
7780 000762          .WORD   16000
7781 000764          .WORD   600
7782 000766          .WORD   74
7783 000770          .WORD   0
7784
7785 000772  T37L2:  .WORD   CONCM,0
7786 000776  T37L3:  .WORD   SBC
7787 001000          ENDDAT
                    ;-------------------------------------------------------------------------------
7788
7789
7808
7809 001000          NEWTST  <ACCELERATOR VBUS INTEGRITY>,TESTD,LOGICD,ERRD,SYNC
```

```
                        .SBTTL  T3D     ACCELERATOR VBUS INTEGRITY
                        ;;*****************************************************************************
                        ;++
                        ;TEST   3D      ACCELERATOR VBUS INTEGRITY
                        ;
                        ; TEST DESCRIPTION
                        ;       THIS TEST CHECKS THE VBUS FOR THE FLOATING POINT ACCELERATOR
                        ;       IF IT IS PRESENT. IF IT IS NOT PRESENT, THIS TEST DOES NOTHING.
                        ;       THE PRESENCE OF THE ACCELERATOR IS DETERMINED BY READING THE
                        ;       ACCELERATOR STATUS REGISTER TO SEE IF IT CONTAINS A 1 IN BITS
                        ;       3 THRU 0.
                        ;
                        ; LOGIC DESCRIPTION
                        ;       THIS TEST CHECKS THE VBUS LOGIC ON THE FPA.
                        ;
                        ; ERROR DESCRIPTION
                        ;       DATA: EXPECTED CONTENTS OF VBUS CONTROL REGISTER
                        ;             RECEIVED CONTENTS OF VBUS CONTROL REGISTER
                        ;             LOOP COUNT -- INDICATES WHICH BIT OF THE VBUS FAILED,
                        ;             I.E. 1 = BIT 0, 2 = BIT 1, 3 = BIT 2, ECT.
                        ;
                        ; SYNC POINT DESCRIPTION
                        ;--
                        ;;*****************************************************************************
            001000      T3D:
   7810 001004                  INITIALIZE
   7811
   7812 001006                  SUBTEST
                        ;;/////////////////////////////////////////////////////////////////////////////
            001006      T3DS1:
   7813                 ;+
   7814                 ; READ THE ACCELERATOR STATUS REGISTER TO SEE IF ITS PRESENT
   7815                 ;-
   7816
   7817 001010                  LOADCA  IDCS,T3DL1      ; LOAD THE REGISTER ADDRESS
   7818 001020                  CLOCK   2               ; GO TO CPT2
   7819 001024                  CMPCAM  EQ,IDDATLO,T3DM1,,T3DL2 ; CHECK FOR FPA
   7820 001040                  SKIPERROR               ; SKIP IF NO FPA PRESENT
   7821
   7822 001044                  SUBTEST
                        ;;/////////////////////////////////////////////////////////////////////////////
            001044      T3DS2:
   7823                 ;+
   7824                 ; LOAD THE FPA CHANNEL WITH ALL ZERO'S AND CHECK THAT IT GOT LOADED
   7825                 ;-
   7826
   7827 001046                  ERLOOP
   7828 001050                  LOOP    I,1,60
   7829 001062                  LOADCA  VBCTRL,T3DL3    ; LOAD A BIT OF THE BUS
   7830 001072                  ENDLOOP I
   7831
   7832 001076                  LOOP    I,1,60
   7833 001110                  CMPCAM  EQ,VBCTRL,T3DM2,,T3DL4 ; CHECK EACH BIT FOR 0
   7834 001124                  IFERROR ,FPAX           ; IF ERROR CALL OUT FPA
   7835 001132                  LOADCA  VBCTRL,T3DL3    ; GET THE NEXT BIT
   7836 001142                  ENDLOOP I
   7837
```

```
                        SUBTEST
                ;////////////////////////////////////////////////////////////////////////////////
        001146  T3DS3:
                ;
                ; LOAD THE FPA CHANNEL WITH ALL ONE'S AND CHECK THAT IT GOT LOADED
                ;

                        ERLOOP
        001150          LOOP    I,1,60
        001152          LOADCA  VBCTRL,T3DL5    ; LOAD THE ONE'S
        001164          ENDLOOP I

        001174          LOOP    I,1,60
        001200          CMPCAM  EQ,VBCTRL,T3DM2,,T3DM2 ; CHECK THAT ONE'S ARE COMMING BACK
        001212          IFERROR ,FPAX
        001226          LOADCA  VBCTRL,T3DL5    ; CLOCK THE BUS
        001234          ENDLOOP I
        001244          SKIP

        001250  FPAX:   REPORT  <FNM,FCT,CIB,VBUS>

        001254          TESTDAT
                ;-------------------------------------------------------------------------------
        001270  T3DL1:  .WORD   IDMAINT+ACCST
        001272  T3DL2:  .WORD   1                       ; INDICATES FPA PRESENT
        001274  T3DL3:  .WORD   VBCLK                   ; VBUS CLOCK BIT
        001276  T3DL4:  .WORD   0                       ; EXPECTED VBUS DATA
        001300  T3DL5:  .WORD   SLFTST+VBCLK            ; SELFTEST AND VBUS CLOCK
        001302  T3DM1:  .WORD   17                      ; ACCST MASK FOR BITS <3:0>
        001304  T3DM2:  .WORD   40000                   ; CHANNEL 6 OF VBUS
                ;  HEXDATA      4000
        001306          ENDDAT
                ;-------------------------------------------------------------------------------

                        FORCEOVR
                .SBTTL  SECTION NUMBER 16
        000000          .PSECT  OVR26
        000034          NEWTST  <RESERVED FOR FUTURE EXPANSION>
```

```
                .SBTTL  T3E     RESERVED FOR FUTURE EXPANSION
                ;;*****************************************************************************
                ;++
                ;TEST   3E      RESERVED FOR FUTURE EXPANSION
                ;--
                ;;*****************************************************************************
     000034     T3E:
  7870          ;       .BLKB   200
  7871 000040           FORCEOVR
                .SBTTL  SECTION NUMBER 17
     000000             .PSECT  OVR27
  7872 000034           NEWTST  <RESERVED FOR FUTURE EXPANSION>
```

```
                        .SBTTL  T3F     RESERVED FOR FUTURE EXPANSION
                        ;;******************************************************************************
                        ;++
                        ;TEST   3F      RESERVED FOR FUTURE EXPANSION
                        ;--
                        ;;******************************************************************************
        000034  T3F:
 7873                   ;       .BLKB   200
 7874 000040            FORCEOVR
                        .SBTTL  SECTION NUMBER 18
        000000          .PSECT  OVR30
 7875                   ;       .BLKB   200
 7876 000034            FORCEOVR
                        .SBTTL  SECTION NUMBER 19
        000000          .PSECT  OVR31
 7877                   ;       .BLKB   200
 7878 000034            FORCEOVR
                        .SBTTL  SECTION NUMBER 1A
        000000          .PSECT  OVR32
 7879                   ;       .BLKB   200
 7880 000034            FORCEOVR
                        .SBTTL  SECTION NUMBER 1B
        000000          .PSECT  OVR33
 7881                   ;       .BLKB   200
 7882 000034            FORCEOVR
                        .SBTTL  SECTION NUMBER 1C
        000000          .PSECT  OVR34
 7883                   ;       .BLKB   200
 7884 000034            FORCEOVR
                        .SBTTL  SECTION NUMBER 1D
        000000          .PSECT  OVR35
 7885                   ;       .BLKB   200
 7886 000034            FORCEOVR
                        .SBTTL  SECTION NUMBER 1E
        000000          .PSECT  OVR36
 7887                   ;       .BLKB   200
 7888 000034            FORCEOVR
                        .SBTTL  SECTION NUMBER 1F
        000000          .PSECT  OVR37
 7889                   ;       .BLKB   200
 7890 000034            ENDHC
 7891                   .END
```

SYMBOL TABLE

```
A     = 000077          CLKSLO= 000020          DP10    002160R     022 GOCHA2= 000012          MIC2FL= 004000
ACCMNT= 000026          CLKSTP= 000040          DP11    000156R     023 GPR1    001040R 025 MIMIC   000410R 011
ACCST = 000027          CLRUWR= 000200          DP11A   000412R     023 GPR1A   001304R 025 MIVB    000440R 011
ACC0  = 000024          CMSKX   001600R 004     DP11B   000646R     023 HALTD = 000001          MI1MIC  000424R 011
ACC1  = 000025          CMSK1   001750R 003     DP11C   001102R     023 HALTI = 000002          MNTRTN= 002000
ADAOFF= 000126          CNVERT= 000007          DP12    000224R     024 HARDC = 000001          MPC   = 000037
B     = 000234R   027   COM   = 100000          DP12A   000334R     024 HCMONI= 000000          MPFC  = 000026
BELL  = 000020          CONACK= 000200          DP12B   000630R     024 IBCLKS= 000012          MPGOCH= 000024
BUSOFF= 000120          CONCM = 001000          DP12C   000740R     024 IBDAT = 000000          MPI   = 000036
BYL   = 000044          CONID = 000003          DP12D   001234R     024 IBICT = 000013          MPS   = 000040
BYU   = 000045          CONMCR= 173032          DP12E   001344R     024 IBNIN = 000011          MSB   = 000022
B1FULL= 000004          CONMCS= 173034          DP2     001462R     015 IBTOD = 000001          MSBE  = 000043
B1INUS= 000400          CONRXD= 000005          DP2A    002044R     015 ICL   = 000012          MSFC  = 000032
B2FULL= 000200          CONRXS= 000004          DP3     000230R     016 ICLID   001206R 004     MSGOCH= 000030
B2INUS= 000040          CONT  = 004000          DP3A    000632R     016 IDBUS = 000051          MSK2    001142R 012
CAM   = 000013          CONTXD= 000007          DP4     001340R     016 IDCS  = 173030          MSK5A   001456R 003
CCPT0 = 000200          CONTXS= 000006          DP4A    001722R     016 IDCYCL= 100000          M256K0= 000116
CCPT1 = 000100          CPURUN= 000400          DP5     000214R     017 IDDATH= 173010          M4KOFF= 000052
CCPT2 = 000040          CSBUS = 000050          DP5A    000616R     017 IDDATL= 173006          M6KOFF= 000072
CCPT3 = 000020          CSERR   001434R 007     DP6     000230R     020 IDMAIN= 000200          M64KOF= 000114
CDM   = 000014          CTMP1   000732R 003     DP6A    000220R     020 IDP   = 000021          NBYTE0  002336R 015
CEH   = 000011          CTMP10  001470R 004     DP6B    000776R     020 IDREGH= 000001          NBYTE1  002340R 015
CES   = 000014          CTMP11  001556R 004     DP6C    001032R     020 IDREGL= 177777          NER   = 000010
CHKSWI= 000023          CTMP12  001726R 004     DP6D    000220R     021 IDWRIT= 000100          NE.   = 000004
CIA   = 000056          CTMP13  002254R 004     DP6E    000230R     021 ID1     001642R 005     OBYTE0  001174R 016
CIB   = 000000          CTMP14  002260R 004     DP7     001362R     022 INIT  = 010000          OBYTE1  001176R 016
CIBT00  000340R   003   CTMP16  001342R 005     DRA   = 000055          IRC   = 000020          OPENFL= 000003
CIBX4   001546R   003   CTMP17  000716R 005     DUM1  = 177777          IRCID   001220R 004     OPNFL1= 000011
CIB1    000716R   003   CTMP2   001106R 003     D.SV  = 000056          ISP   = 000054          PARSER= 000010
CIB10   001712R   004   CTMP20  000726R 005     D0    = 000001          ITSTPT= 000004          PBYTE0  002214R 016
CIB11   002040R   004   CTMP3   001112R 003     D1    = 000017          KEYERR= 020000          PBYTE1  002216R 016
CIB12   000242R   005   CTMP4   001710R 003     EQ.   = 000000          KEYQUE= 010000          PCBB  = 000072
CIB12A  000252R   005   CTMP5   001714R 003     ERABT = 000040          KSP   = 000050          PCS   = 000003
CIB13   000702R   005   CTMP7   002266R 003     ESP   = 000051          LCANWC= 000014          PCSERR  000114R 011
CIB14   001326R   005   CTMP7X  001166R 004     FAD   = 000033          LCWRON= 000016          PROCEE= 000001
CIB3    001072R   003   CTMP8X  001570R 004     FAILCH= 000016          LDCONS= 000021          PSL   = 000017
CIB3A   000774R   003   CTMP96  000272R 005     FCHAI1= 000020          LOADCN= 000006          POBR  = 000044
CIB30   002562R   005   CTMP97  000270R 005     FCHAI2= 000022          LOOP  = 000004          POLR  = 000074
CIB32   000374R   006   CTMP98  001116R 003     FCT   = 000034          LOPFI = 000000          P1BR  = 000045
CIB33   000200R   027   CTMP99  000266R 005     FETADR= 000662R 027     LOPFJ = 000000          P1LR  = 000075
CIB34   000346R   027   CTRLC = 040000          FLPYMS= 003000          LOPFK = 000000          QBYTE0  001160R 017
CIB35   000526R   027   CX1TMP  002264R 004     FLPYON= 010000          LOSS  = 000100          QBYTE1  001162R 017
CIB36   000400R   004   CX2TMP  002266R 004     FLPY2 = 001000          LOST  = 000200          Q.SV  = 000057
CIB37   000726R   027   CX3TMP  002272R 004     FLPY3 = 002000          M     = 000000          RADGET= 000010
CIB4    001650R   003   DAP   = 000004          FLPY4 = 003000          MAT   = 000041          RADHEX= 000020
CIB4A   001674R   003   DAPCTL  000776R 022     FMH   = 000031          MAY   = 000025          RADOCT= 000010
CIB5    002252R   003   DBP   = 000010          FMIDHI= 173026          MAY4  = 000046          RARE    001456R 026
CIB5A   001432R   003   DCP   = 000005          FMIDLO= 173024          MAY6  = 000035          RARW    001562R 025
CIB6    001006R   004   DDP   = 000006          FML   = 000032          MAY8  = 000047          RAWR    001426R 026
CIB6A   001032R   004   DEP   = 000007          FNM   = 000030          MBA   = 000054          RBRE    001472R 026
CIB6B   001056R   004   DICMD = 001000          FPA   = 010000          MBYTE0  001306R 015     RBRW    001612R 025
CIB6C   001102R   004   DIRECT= 000014          FPAX    001254R 027     MBYTE1  001310R 015     RBWR    001426R 026
CIB6D   001126R   004   DIRERR= 000100          FPDA  = 000055          MCN   = 000023          RBYTE0  002140R 020
CIB6E   001152R   004   DIRPTR= 000000R 002     FR0   = 000010          MDMTYP= 000022          RBYTE1  002142R 020
CIB7    001454R   004   DNEIE = 000040          FR1   = 000020          MDT   = 000024          RCRE    001506R 026
CLK   = 000026          DP1     000364R 015     GOCHAI= 000004          MICINI  000572R 006     RCRW    001642R 025
CLKFST= 00001C          DP1A    000774R 015     GOCHA1= 000006          MIC1FL= 002000          RCWR    001442R 026
```

SYMBOL TABLE

```
RDYIE = 000100          SYNC0C  002156R     003 SYNC35  000730R     006 SYNC71  000726R     015 TEMP8 = 000070
READSC= 000004          SYNC0D  002166R     003 SYNC36  001124R     006 SYNC72  001410R     015 TEMP9 = 000071
RMWRON= 000015          SYNC0E  002212R     003 SYNC37  001174R     006 SYNC73  001772R     015 TESTST= 000002
ROM0  = 173000          SYNC0F  002222R     003 SYNC38  000464R     006 SYNC74  000156R     016 TINIT = 000000
ROM1  = 173002          SYNC00  000532R     003 SYNC39  001314R     006 SYNC75  000560R     016 TMERTR= 000017
RPTUSC  001042R     006 SYNC01  000542R     003 SYNC4A  000226R     007 SYNC76  001266R     016 TMPX1   000566R     007
RUNFLG= 000002          SYNC02  000552R     003 SYNC4B  000364R     007 SYNC77  001650R     016 TMPX24  000670R     007
RWDQ  = 000010          SYNC03  000634R     003 SYNC4C  001316R     007 SYNC78  000142R     017 TMPX25  001644R     007
RXDNE = 173014          SYNC04  000644R     003 SYNC4D  001560R     007 SYNC79  000544R     017 TMPX33  002326R     012
RXVEC = 000304          SYNC05  000654R     003 SYNC4E  002060R     007 SYNC8A  001210R     024 TMPX34  002430R     012
R$SET = 000020          SYNC06  001022R     003 SYNC4F  000130R     010 SYNC8B  000122R     025 TMPX42  001136R     012
R6    =%000006          SYNC07  001032R     003 SYNC40  001564R     006 SYNC8C  000126R     025 TMPX52  002204R     013
R7    =%000007          SYNC08  001504R     003 SYNC41  001626R     006 SYNC8D  000240R     025 TMPX60  001102R     016
SBC   = 000002          SYNC09  001514R     003 SYNC42  001652R     006 SYNC8E  000244R     025 TMPX61  001166R     016
SBH   = 000017          SYNC1A  001300R     004 SYNC43  002024R     006 SYNC8F  000372R     025 TMP1    001056R     006
SBHID   001224R     004 SYNC1B  001340R     004 SYNC44  002044R     006 SYNC80  002106R     022 TMP10   002616R     006
SBICP = 000036          SYNC1C  001632R     004 SYNC45  002140R     006 SYNC81  000126R     023 TMP100  001610R     007
SBIERR= 000031          SYNC1D  001660R     004 SYNC46  002160R     006 SYNC82  000362R     C23 TMP101  001640R     007
SBIFLT= 000033          SYNC1E  001770R     004 SYNC47  002256R     006 SYNC83  000616R     023 TMP102  001650R     007
SBIMAT= 000035          SYNC1F  001776R     004 SYNC48  002356R     006 SYNC84  001052R     023 TMP103  001704R     007
SBISCM= 000034          SYNC10  000624R     004 SYNC49  000066R     007 SYNC85  000126R     024 TMP11   002622R     006
SBISIL= 000030          SYNC11  000744R     004 SYNC5A  000672R     012 SYNC86  000200R     024 TMP110  001646R     022
SBITO = 000032          SYNC12  001240R     003 SYNC5B  000774R     012 SYNC87  000532R     024 TMP111  001662R     022
SBL   = 000016          SYNC13  001266R     003 SYNC5C  001232R     012 SYNC88  000604R     024 TMP112  001706R     022
SBR   = 000046          SYNC14  001310R     003 SYNC5D  000120R     013 SYNC89  001136R     024 TMP113  002174R     022
SBYTE0  001714R     022 SYNC15  001342R     003 SYNC5E  000130R     013 SYNC9A  001002R     006 TMP114  002254R     022
SBYTE1  001716R     022 SYNC16  001370R     003 SYNC5F  000244R     013 SYNC9B  001074R     013 TMP115  002224R     022
SCBB  = 000073          SYNC17  001412R     003 SYNC50  000164R     010 SYNC9C  000572R     005 TMP116  001242R     023
SCTSPA= 040000          SYNC18  001260R     004 SYNC51  000070R     011 SYNC9D  001216R     005 TMP117  001272R     023
SINST = 000400          SYNC19  001264R     004 SYNC52  000204R     011 SYNC9F  000122R     027 TMP12   002624R     006
SIR   = 000016          SYNC2A  000436R     005 SYNC53  002076R     012 SYNC90  000376R     025 TMP120  001274R     023
SLFTST= 000004          SYNC2B  000534R     005 SYNC54  002106R     012 SYNC91  000510R     025 TMP121  001276R     023
SLR   = 000076          SYNC2C  000776R     005 SYNC55  002236R     012 SYNC92  000514R     025 TMP122  001300R     023
SOMM  = 000100          SYNC2D  001004R     005 SYNC56  002246R     012 SYNC93  000644R     025 TMP123  001460R     024
SPARE1= 173004          SYNC2E  001062R     005 SYNC57  000242R     012 SYNC94  000650R     025 TMP124  001510R     024
SPARE2= 173012          SYNC2F  001160R     005 SYNC59  000520R     012 SYNC95  000762R     025 TMP125  001524R     024
SSP   = 000052          SYNC20  002074R     004 SYNC6A  002124R     013 SYNC96  000766R     025 TMP126  001540R     024
STOVR = 000000R     041 SYNC21  002102R     004 SYNC6C  000134R     015 SYNC97  000474R     004 TMP127  001544R     024
STS   = 000004          SYNC22  002206R     004 SYNC6D  000150R     015 SYNC98  001170R     003 TMP13   000142R     007
SYNCAA  000674R     027 SYNC23  000044R     005 SYNC6E  000154R     015 SYNC99  001364R     004 TMP130  001550R     024
SYNCAB  001422R     012 SYNC24  000104R     005 SYNC6F  000706R     015 TBDAT = 000020          TMP131  001554R     024
SYNCAC  001502R     012 SYNC25  000124R     005 SYNC60  000254R     013 TBER0 = 000022          TMP14   000276R     007
SYNCAD  001562R     012 SYNC26  000166R     005 SYNC61  000470R     013 TBER1 = 000023          TMP2    001246R     006
SYNCAE  001646R     012 SYNC27  000216R     005 SYNC62  000570R     013 TBLLOC= 000010R     041 TMP201  002170R     013
SYNCAF  000130R     014 SYNC28  000352R     005 SYNC63  000650R     013 TBM   = 000015          TMP23   000466R     007
SYNCA0  000632R     005 SYNC29  000360R     005 SYNC64  000742R     013 TBMID   001214R     004 TMP24   000570R     007
SYNCA1  001256R     005 SYNC3A  001322R     006 SYNC65  001026R     013 TBYTE0  001672R     025 TMP25   000770R     007
SYNCA2  000062R     004 SYNC3B  001400R     006 SYNC66  001316R     013 TBYTE1  001674R     025 TMP26   002140R     007
SYNCA3  000136R     004 SYNC3C  001406R     006 SYNC67  001414R     013 TEMP  = 000214R     002 TMP27   002120R     007
SYNCA4  000204R     004 SYNC3D  001524R     006 SYNC68  001564R     013 TEMP0 = 000060          TMP3    001250R     006
SYNCA5  000260R     004 SYNC3E  001532R     006 SYNC69  002000R     013 TEMP1 = 000600R     027 TMP30   002200R     007
SYNCA6  000346R     004 SYNC3F  001540R     006 SYNC7A  000126R     020 TEMP2 = 000610R     027 TMP31   000230R     010
SYNCA7  000310R     027 SYNC30  001434R     005 SYNC7B  000704R     020 TEMP3 = 000063          TMP32   000360R     010
SYNCA8  000470R     027 SYNC31  001534R     005 SYNC7C  000620R     021 TEMP4 = 000064          TMP33   002276R     012
SYNCA9  000624R     027 SYNC32  000640R     006 SYNC7D  001310R     022 TEMP5 = 000065          TMP34   002402R     012
SYNC0A  001574R     003 SYNC33  000646R     006 SYNC7F  001770R     022 TEMP6 = 000066          TMP34Z  002504R     012
SYNC0B  001604R     003 SYNC34  000722R     006 SYNC70  000722R     015 TEMP7 = 000067          TMP35   000314R     012
```

SYMBOL TABLE

```
TMP35A  000306R      012 TWRITE= 000001          T06S1   002140R      003 T12L1   000522R      006 T2CS2   001726R      015
TMP35B  000312R      012 TXRDY = 173016          T06S2   002206R      003 T12L2   000524R      006 T2D     000034R      016
TMP36   000354R      012 TXVEC = 000300          T07     000034R      004 T13     000526R      006 T2DL0   001302R      023
TMP37   001076R      012 TYP1  = 000012          T07L1   001602R      004 T13EE   002302R      004 T2DL1   001304R      023
TMP37A  001050R      012 TYP2  = 000013          T07L2   001604R      004 T14     000606R      006 T2DL2   001310R      023
TMP37B  001140R      012 TOA     001612R     004 T07L3   001606R      004 T14EE   000316R      005 T2DL3   001312R      023
TMP4    001256R      006 TOAL2   000314R     005 T07L4   001610R      004 T14S1   000614R      006 T2DS1   000072R      016
TMP40   001102R      012 TOB     001736R     004 T07S1   000042R      004 T14S2   000676R      006 T2DS2   000474R      016
TMP41   001106R      012 TOBL1   000730R     005 T07S2   000102R      004 T14S3   000760R      006 T2E     001200R      016
TMP42   001122R      012 TOBL2   000736R     005 T07S3   000164R      004 T15     001072R      006 T2EE    000736R      003
TMP43   001326R      012 TOBL3   000740R     005 T07S4   000224R      004 T15EE   000742R      005 T2ES1   001222R      016
TMP44   001332R      012 TOBS1   001744R     004 T07S5   000306R      004 T15S1   001100R      006 T2ES2   001604R      016
TMP45   001206R      013 TOBS2   002050R     004 T08     000430R      004 T15S2   001162R      006 T2F     000034R      017
TMP46   001212R      013 TOBS3   002140R     004 T08L1   001732R      004 T16     001260R      006 T2FL1   002114R      020
TMP47   001216R      013 TOC     000034R     005 T08M1   001734R      004 T16EE   001372R      005 T2FL6   002116R      020
TMP5    001502R      006 TOCL1   001344R     005 T08S1   000434R      004 T16L1   000304R      007 T2FS1   000056R      017
TMP5A   001446R      003 TOCL2   001352R     005 T08S2   000542R      004 T16S1   001266R      006 T2FS2   000460R      017
TMP5B   001452R      003 TOCL3   001356R     005 T08S3   000662R      004 T16S2   001352R      006 T2FV1   002120R      020
TMP5C   001454R      003 TOCL4   001366R     005 T09     001230R      004 T16V1   001506R      006 T2FV2   002124R      020
TMP50   001460R      013 TOCL5   001370R     005 T09L1   002276R      004 T17     001512R      006 T2FV3   002130R      020
TMP500  001554R      025 TOD     000316R     005 T09S1   001236R      004 T17EE   002130R      005 T2FV4   002134R      020
TMP501  001560R      025 TODS1   000324R     005 T09S2   001326R      004 T17L1   002660R      006 T20     000034R      012
TMP51   001464R      013 TODS2   000422R     005 T09S3   001360R      004 T17L2   002674R      006 T20EE   002626R      005
TMP510  001660R      005 TODS3   000474R     005 T1A     000150R      007 T17S1   001516R      006 T20L1   001334R      012
TMP511  001750R      005 TODS4   000560R     005 T1B     000306R      007 T17S2   001610R      006 T20L10  000302R      012
TMP512  002040R      005 TODS5   000620R     005 T1BL1   000510R      010 T17V2   002630R      006 T20L2   001350R      012
TMP513  001656R      005 TOE     000742R     005 T1BL2   000514R      010 T17V21  002634R      006 T20L3   001352R      012
TMP52   001474R      013 TOES1   000760R     005 T1BV1   001170R      007 T17V22  002644R      006 T20M1   001354R      012
TMP53   001244R      015 TOES2   001046R     005 T1C     001174R      007 T17V3   002654R      006 T20S1   000042R      012
TMP54   001274R      015 TOES3   001120R     005 T1CS1   001202R      007 T18     001766R      006 T21     000156R      012
TMP55   001276R      015 TOES4   001204R     005 T1CS2   001444R      007 T18S1   001774R      006 T21EE   000442R      006
TMP55A  001302R      015 TOES5   001244R     005 T1D     001776R      007 T18S2   002064R      006 T21L1   000364R      013
TMP56   001260R      015 TOF     001372R     005 T1DL1   001654R      007 T18S3   002200R      006 T21L2   000370R      013
TMP57   002314R      015 TOOL1   000476R     003 T1DL2   001714R      007 T18S4   002302R      006 T21L3   000372R      013
TMP6    001722R      006 TO1     000034R     003 T1DV1   001716R      007 T19     000034R      007 T21L4   000374R      013
TMP60   001132R      016 TO1S1   000040R     003 T1E     000034R      010 T2A     000034R      014 T21L5   000376R      013
TMP61   001162R      016 TO1S2   000256R     003 T1EE    000500R      003 T2AD1   001012R      022 T21M1   000400R      013
TMP62   002172R      016 TO1S3   000350R     003 T1F     000034R      011 T2AD2   001016R      022 T22     000444R      012
TMP63   001066R      017 TO2     000500R     003 T1FS1   000042R      011 T2AD3   001022R      022 T22EE   000526R      006
TMP65   001146R      017 TO2S1   000506R     003 T1FS2   000124R      011 T2AD4   001026R      022 T22L1   001222R      013
TMP66   001152R      017 TO2S2   000610R     003 T10     002130R      005 T2AD5   001032R      022 T22L2   001226R      013
TMP67   001156R      017 TO3     000736R     003 T10EE   001230R      004 T2AL1   001304R      015 T22L3   001242R      013
TMP7    001726R      006 TO3L1   001120R     003 T10L1   001062R      006 T2AM1   001036R      022 T22L4   001246R      013
TMP70   001244R      020 TO3S1   000744R     003 T10L2   001064R      006 T2AM2   001132R      022 T22L5   001250R      013
TMP71   001310R      020 TO3S2   001004R     003 T10L3   001066R      006 T2AM3   001146R      022 T22L6   001252R      013
TMP72   001510R      020 TO4     001122R     003 T10L4   001070R      006 T2AM4   001162R      022 T22L7   001254R      013
TMP73   001710R      020 TO4L1   001752R     003 T10S1   002136R      005 T2AM5   001176R      022 T22L8   001256R      013
TMP74   001714R      020 TO4S1   001130R     003 T10S2   002176R      005 T2AM6   001212R      022 T22L9   001260R      013
TMP75   002014R      020 TO4S2   001222R     003 T10S3   002256R      005 T2AV1   001312R      015 T22M1   001244R      013
TOIDHI= 173022          TO4S3   001324R     003 T10S4   002316R      005 T2AV2   001316R      015 T22S1   000452R      012
TOIDLO= 173020          TO5     001460R     003 T10S5   002412R      005 T2B     000034R      015 T22S2   000574R      012
TPCINI= 000034          TO5L1   001170R     004 T10S6   002462R      005 T2BL1   001712R      022 T22S3   000726R      012
TRAPVE= 000034          TO5L2   001174R     004 T10S7   002520R      005 T2BS1   000056R      015 T23     001144R      012
TREAD = 000002          TO5L3   001200R     004 T11     000034R      006 T2BS2   000630R      015 T23EE   000606R      006
TRS   = 000027          TO5S1   001466R     003 T11EE   001612R      004 T2C     001322R      015 T23L1   001500R      013
TSTMFG= 000024          TO5S2   001556R     003 T12     000442R      006 T2CL1   002260R      022 T23L2   001502R      013
TSTSPA= 020000          TO6     002132R     003 T12EE   001736R      004 T2CS1   001344R      015 T24     001356R      012
```

SYMBOL TABLE

T24EE	001072R	006	T3DL5	001300R	027	T34	001720R	022	T38S1	000116R	026	USC11	002136R	012	
T24L1	002232R	013	T3DM1	001302R	027	T34EE	001776R	007	T38S2	000322R	026	USC12	001312R	012	
T24L2	002236R	013	T3DM2	001304R	027	T34L1	000362R	027	T38S3	000612R	026	USC13	000164R	013	
T24L3	002240R	013	T3DS1	001006R	027	T34L2	000412R	027	T39	000034R	027	USC14	001172R	013	
T24L4	002254R	013	T3DS2	001044R	027	T34L3	000414R	027	T39L1	000200R	014	USC14A	001162R	013	
T24M1	002230R	013	T3DS3	001146R	027	T34L4	000416R	027	T39L2	000214R	014	USC15	001444R	013	
T24S1	001364R	012	T3E	000034R	030	T34L5	000420R	027	T39L3	000216R	014	USC16	002154R	013	
T24S2	001442R	012	T3EE	001122R	003	T34S1	001726R	022	T39L4	000220R	014	USC2	001232R	006	
T24S3	001522R	012	T3F	000034R	031	T34S2	002026R	022	T39L5	000222R	014	USC3	001442R	006	
T24S4	001622R	012	T30	000034R	020	T34V1	001556R	024	T4EE	001460R	003	USC38	001702R	012	
T25	002000R	012	T30EE	002676R	006	T34V1A	001562R	024	T40EE	000156R	012	USC39	000154R	014	
T25EE	001260R	006	T30L1	002576R	005	T34V2	001566R	024	T40L1	000136R	012	USC39A	000164R	014	
T25S1	002006R	012	T30L2	002600R	005	T34V2A	001572R	024	T40L2	000152R	012	USC4	001706R	006	
T25S2	002146R	012	T30L3	002604R	005	T34V3	001576R	024	T40L3	000154R	012	USC40	000122R	012	
T26	000034R	013	T30L4	002606R	005	T34V3A	001602R	024	T41EE	000444R	012	USC5	002406R	006	
T26EE	001512R	006	T30L5	002612R	005	T34V5	001606R	024	T42EE	001144R	012	USC5A	002446R	006	
T26S1	000042R	013	T30M1	002614R	005	T34V6	002004R	024	T43EE	001356R	012	USC5D	002472R	006	
T26S2	000174R	013	T30M2	002616R	005	T35	000034R	023	T44EE	002000R	012	USC5E	002502R	006	
T26S3	000304R	013	T30M3	002620R	005	T35EE	002300R	007	T45EE	002506R	012	USC6	000126R	007	
T27	000402R	013	T30M4	002622R	005	T35L1	000542R	027	T46EE	000402R	013	USC7	000262R	007	
T27EE	001766R	006	T30M5	002624R	005	T35L2	000556R	027	T47EE	001262R	013	USP =	000053		
T27L1	001172R	016	T30S1	000056R	020	T35L3	000560R	027	T5EE	002132R	003	VBCLK =	000001		
T27S1	000410R	013	T30S2	000634R	020	T35L4	000562R	027	T50EE	001504R	013	VBCTRL=	173036		
T27S2	000510R	013	T31	000034R	021	T35S1	000042R	023	T51EE	002276R	013	VBLOAD=	000002		
T27S3	000610R	013	T31EE	000150R	007	T35S2	000312R	023	T52EE	000234R	014	VBMNT1	000274R	005	
T27S4	000704R	013	T31S1	000056R	021	T35S3	000546R	023	T53EE	001322R	015	VBMNT2	000300R	005	
T27S5	000756R	013	T31S2	000436R	021	T35S4	001002R	023	T54EE	002342R	015	VBMNT3	000304R	005	
T27S6	001042R	013	T31S3	000536R	021	T36	000034R	024	T55EE	001200R	016	VBMNT4	000310R	005	
T28	001262R	013	T32	000034R	022	T36EE	000524R	010	T56EE	002220R	016	VBUS =	000052		
T28L1	002206R	016	T32EE	000306R	007	T36L1	000414R	004	T57EE	001164R	017	VB1A	002516R	006	
T28L2	002212R	016	T32L1	000410R	006	T36L2	000420R	004	T6EE	002274R	003	VB1B	002556R	006	
T28S1	001270R	013	T32L2	000434R	006	T36L3	000422R	004	T60EE	002144R	020	VECT =	000015		
T28S2	001336R	013	T32L3	000436R	006	T36L4	000424R	004	T61EE	001236R	021	WCS =	000002		
T28V1	002260R	013	T32M1	000440R	006	T36L5	000426R	004	T62EE	001226R	022	WCS1	000374R	011	
T28V2	002264R	013	T32S1	000042R	022	T36S1	000042R	024	T63EE	001720R	022	WCS2K =	000042		
T28V3	002272R	013	T32S2	000156R	022	T36S2	000444R	024	T64EE	002262R	022	WRITSC=	000005		
T29	001504R	013	T32S3	000274R	022	T36S3	001050R	024	T65EE	001446R	023	X =	000164		
T29S1	001512R	013	T32S4	000412R	022	T37	000034R	025	T66EE	002022R	024	XCIB6A	000532R	004	
T29S2	001750R	013	T32S5	000536R	022	T37EE	000450R	011	T67EE	001676R	025	XTMP10	000742R	021	
T29S3	002054R	013	T32S6	000654R	022	T37L1	000742R	027	T7EE	000430R	004	XTMP12	001546R	024	
T3A	000242R	027	T33	001226R	022	T37L2	000772R	027	T70EE	001552R	026	XTMP17	001162R	021	
T3AF1	001032R	026	T33EE	001174R	007	T37L3	000776R	027	T71EE	000242R	027	XTMP34	002500R	012	
T3AL1	001326R	026	T33L1	000214R	027	T37S1	000056R	025	T72EE	000422R	027	XTMP52	002210R	013	
T3AL2	001522R	026	T33L2	000230R	027	T37S2	000174R	025	T73EE	000564R	027	XTMP70	000676R	021	
T3AL3	001526R	026	T33L3	000232R	027	T37S3	000326R	025	T74EE	001000R	027	XTMP73	000756R	021	
T3AL4	001532R	026	T33L4	000236R	027	T37S4	000444R	025	T75EE	001306R	027	XT2FL1	001176R	021	
T3AL5	001546R	026	T33L5	000240R	027	T37S5	000562R	025	UBA =	000053		XT2FL2	000762R	021	
T3AL6	001550R	026	T33S1	001234R	022	T37S6	000716R	025	UBYTE0	001232R	021	XT2FL3	001062R	021	
T3B	000422R	027	T33V1	001316R	023	T38	000034R	026	UBYTE1	001234R	021	XT2FL4	001200R	021	
T3C	000564R	027	T33V2	001326R	023	T38L1	001716R	012	UPC12 =	001000		XT2FL5	001204R	021	
T3CS1	000572R	027	T33V2A	001340R	023	T38L2	001732R	012	USC =	000001		XT2FL6	001210R	021	
T3CS2	000652R	027	T33V2B	001352R	023	T38L3	001734R	012	USCADR=	000042		XT2FV1	001212R	021	
T3D	001000R	027	T33V2C	001364R	023	T38L4	001736R	012	USCBRK=	000041		XT2FV2	001216R	021	
T3DL1	001270R	027	T33V3	001376R	023	T38L5	001742R	012	USCDAT=	000043		XT2FV3	001222R	021	
T3DL2	001272R	027	T33V3A	001410R	023	T38L6	001756R	012	USCID	001202R	004	XT2FV4	001226R	021	
T3DL3	001274R	027	T33V3B	001422R	023	T38L7	001772R	012	USCSTK=	000040		XX57	002332R	015	
T3DL4	001276R	027	T33V3C	001434R	023	T38L8	001776R	012	USC1	001034R	012	X52	002224R	013	

SYMBOL TABLE

```
X57     002330R     015 $FNF  = 000002          $NEWTS= 000005          $STN  = 000001          $TER  = 000007
Y     = 000001          $FNR  = 000003          $SECTO= 000263          $TBSY = 000005          $TN   = 000100
Z     = 000136          $FOR  = 000004          $SN   = 000037          $TCTC = 000006          $$SN  = 000036
$FER  = 000001

. ABS.  000000      000
        000000      001
OVR00   000400      002
OVR01   002400      003
OVR02   002400      004
OVR03   003000      005
OVR04   003000      006
OVR05   002400      007
OVR06   000600      010
OVR07   000600      011
OVR10   002600      012
OVR11   002400      013
OVR12   000400      014
OVR13   002400      015
OVR14   002400      016
OVR15   001200      017
OVR16   002200      020
OVR17   001400      021
OVR20   002400      022
OVR21   001600      023
OVR22   002200      024
OVR23   002000      025
OVR24   001600      026
OVR25   001400      027
OVR26   000200      030
OVR27   000200      031
OVR30   000200      032
OVR31   000200      033
OVR32   000200      034
OVR33   000200      035
OVR34   000200      036
OVR35   000200      037
OVR36   000200      040
OVR37   000200      041
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  31846 WORDS  ( 125 PAGES)
DYNAMIC MEMORY:  20060 WORDS  ( 77 PAGES)
ELAPSED TIME:  00:12:05
ESKAD,ESKAD/-SP=ESKABMAC.MLB/ML,ESKAD
```